On the complexity of partitioning a graph into a few connected subgraphs

Julien Bensmail

Noname manuscript No.
(will be inserted by the editor)

On the complexity of partitioning a graph into a few connected subgraphs

Julien Bensmail

the date of receipt and acceptance should be inserted later

Abstract Given a graph G, a sequence $\tau = (n_1, ..., n_p)$ of positive integers summing up to $|V(G)|$ is said to be realizable in G if there exists a realization of τ in G, i.e. a partition $(V_1, ..., V_p)$ of $V(G)$ such that each V_i induces a connected subgraph of G on n_i vertices. We first give a reduction showing that the problem of deciding whether a sequence with c elements is realizable in a graph is NP-complete for every fixed $c \geq 2$. Thanks to slight modifications of this reduction, we then prove additional hardness results on decision problems derived from the previous one. In particular, we show that the previous problem remains NP-complete when a constant number of vertex-membership constraints must be satisfied. We then prove the tightness of an easiness result proved independently by Györi and Lovász regarding a similar problem. We finally show that another graph partition problem, asking whether several partial realizations of τ in G can be extended to obtain whole realizations of τ in G, is Π_2^p-complete.

Keywords arbitrarily partitionable graphs · partition into connected subgraphs · partition under vertex prescriptions · complexity · polynomial hierarchy

1 Introduction

Let G be a connected graph. A sequence $\tau = (n_1, ..., n_p)$ of positive integers is *admissible for* G if $\sum_{i=1}^{p} n_i = |V(G)|$. We say that τ is *realizable in* G if τ is admissible for G and there exists a *realization of* τ *in* G, i.e. a partition $(V_1, ..., V_p)$ of $V(G)$ such that V_i induces a connected subgraph of G on n_i

J. Bensmail
Univ. Bordeaux, LaBRI, UMR 5800, F-33400 Talence, France
CNRS, LaBRI, UMR 5800, F-33400 Talence, France
E-mail: julien.bensmail@labri.fr
Tel.: +33-(0)5-40-00-35-17, Fax: +33-(0)5-40-00-66-69

vertices for every $i \in \{1, ..., p\}$. We refer to p as the *size* of τ. A *p-sequence* is a sequence with size p.

The problem of finding a realization of a given sequence in a graph gained a lot of attention since the result, proved independently by Györi and Lovász, which states that, for any $q \geq 1$, every sequence with size at most q admissible for a q-connected graph G is realizable in G [6,7]. Since then, several graph properties based on the definition above have been investigated.

For instance, we say that G is *arbitrarily partitionable* (AP for short) if every sequence admissible for G is also realizable in G. For the sake of the upcoming next definitions, let us now consider a *k-prescription of G*, i.e. a sequence $(v_1, ..., v_k)$ of k pairwise distinct vertices of G with $k \leq p$, where p is the size of a sequence τ admissible for G. We say that τ is *realizable in G under* $(v_1, ..., v_k)$ if there exists a realization $(V_1, ..., V_p)$ of τ in G such that $v_i \in V_i$ for every $i \in \{1, ..., k\}$. In other words, a k-prescription is a set of vertices that were chosen to belong to the first k parts of a realization of τ in G. Notice that, in our terminology, the k part sizes associated with these k prescribed vertices are the first k ones of the sequence. Finally, the graph G is said to be *arbitrarily partitionable under k prescriptions* (AP+k for short) if every sequence with size at least k admissible for G is realizable in G under every k-prescription of G. All these definitions were introduced to deal with a practical problem of resource sharing among an arbitrary number of users [1, 3].

In this paper, we consider the computational complexity of some decision problems derived from the definitions above. Thanks to our main reduction given in Section 2, we show that the problem of deciding whether a sequence is realizable in a given graph is NP-complete even when restricted to c-sequences for every fixed $c \geq 2$. This reduction may be related to one reduction from [5], where similar gadgets as ours are used to prove the hardness of a min-max tree partition problem. We then augment our reduction in further sections to show additional complexity results. We first prove in Section 3 that requesting prescriptions while partitioning a graph does not alter the complexity of the problem, and this no matter how many such prescriptions are requested. In Section 4, we investigate the tightness of the well-known easiness result proved independently by Györi and Lovász mentioned above. We finally discuss the complexity of the problems of deciding whether a graph is AP or AP+k in Section 5. We locate these two problems in the Π_2^p complexity class of the polynomial hierarchy and explain why we cannot modify our previous reductions to prove that these problems are Π_2^p-complete. We however show that our graph partition problem is not "incompatible" with the notion of Π_2^p-complete problems by pointing out one such Π_2^p-complete problem.

2 Complexity of partitioning a graph into a few connected subgraphs

In this section, we focus on the following decision problem.

Realizable Sequence - RealSeq
Instance: A graph G and a sequence τ.
Question: Is τ realizable in G?

Assuming that the size of τ is constant, we get the following refinement.

Realizable k-Sequence - k-RealSeq
Instance: A graph G and a k-sequence τ.
Question: Is τ realizable in G?

It is already known that RealSeq is computationally hard, even under restrictions on G or τ. In particular, this problem remains NP-complete even when G is a tree with maximum degree 3, or $\tau = (k, ..., k)$ has only one integer value $k \geq 3$ that divides $|V(G)|$ (see [2] and [4], respectively). However, the complexity reductions used to show these restrictions on RealSeq do not imply the existence of a constant threshold $c \geq 1$ such that:

- k-RealSeq is in P for every $k \leq c - 1$;
- k-RealSeq is NP-complete otherwise.

The answer to an instance of 1-RealSeq is *yes* if and only if G is connected. Since the connectedness of a graph can be checked easily, we have $c \geq 2$ assuming that c exists. In what follows, we prove that $c = 2$, i.e. that k-RealSeq is NP-complete for every $k \geq 2$. Our reduction is from the following variant of 3SAT.

1-in-3 SAT
Instance: A 3CNF formula F over variables $\{x_1, ..., x_n\}$ and clauses $\{C_1, ..., C_m\}$.
Question: Is there a 1-in-3 truth assignment of the variables of F, i.e. a truth assignment such that each clause of F has exactly one true literal?

Notice that we can suppose that every possible literal appears in F. Indeed, if x_i does not appear in any clause of F, then the 3CNF formula $F' = F \wedge (x_i \vee \overline{x_i} \vee x_{n+1}) \wedge (x_{n+1} \vee x_{n+1} \vee \overline{x_{n+1}})$, where x_{n+1} is a new variable, admits a 1-in-3 truth assignment of its variables if and only if F admits one too. Since there are $2n$ literals related to the variables of F, a formula equivalent to F that contains every possible literal over its variables can be obtained from F in polynomial time.

Our proof of the NP-completeness of k-RealSeq for every $k \geq 2$ reads as follows. We first show in Theorem 1 below that 2-RealSeq is NP-complete by reduction from 1-in-3 SAT. We then explain, in Theorem 2, how to modify our reduction from 1-in-3 SAT to 2-RealSeq so that we get a reduction from 1-in-3 SAT to k-RealSeq for any $k \geq 3$.

Theorem 1 2-RealSeq *is NP-complete.*

Proof First of all, RealSeq is clearly in NP. One can indeed design an algorithm that takes the graph G, the sequence τ and a realization R of τ in G as input and checks whether R is correct. More precisely, such an algorithm has to check that τ is admissible for G, the parts of R have the correct sizes

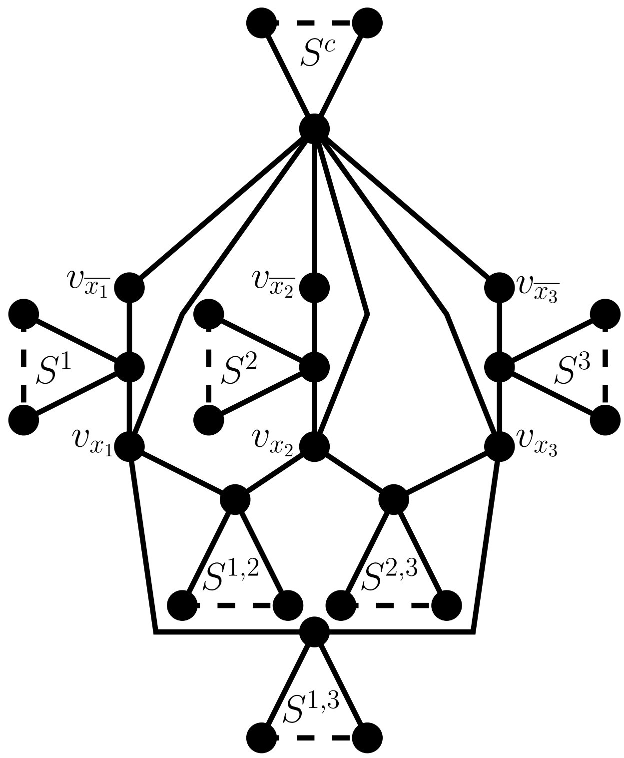

Fig. 1 Resulting subgraph in the clause subgraph of G_F for a clause $C_1 = (x_1 \vee x_2 \vee x_3)$ of F

regarding τ, and that the subgraphs of G induced by R are connected. This verification can be done in polynomial time regardless of the size of τ.

We now prove that 2-RealSeq is NP-complete by reduction from 1-in-3 SAT. For a given formula F over variables $\{x_1, ..., x_n\}$ and clauses $\{C_1, ..., C_m\}$, we construct a graph G_F and a sequence τ_F admissible for G_F such that F is satisfiable in a 1-in-3 way if and only if τ_F is realizable in G_F. Our reduction is performed in such a way that τ_F is a 2-sequence.

The graph G_F is composed of two main vertex-disjoint subgraphs. The first one is the *clause subgraph.* Each literal ℓ_i of F is associated with a *literal vertex* v_{ℓ_i} in the clause subgraph. For each pair of literals ℓ_i and $\overline{\ell_i}$ of F, we then link the literal vertices v_{ℓ_i} and $v_{\overline{\ell_i}}$ to the root vertex of a star S^i with n vertices of degree 1. Two literal vertices v_{ℓ_i} and v_{ℓ_j} such that $\ell_j \neq \overline{\ell_i}$ are similarly linked to the root vertex of a star $S^{i,j}$ with n vertices of degree 1 if they both appear in a same clause of F. We finally add a *control star* S^c with n vertices of degree 1 to the clause subgraph of G_F and link its root to every literal vertex so that the clause subgraph is connected.

The construction so far is detailed in Figure 1. Let n_2 be the number of vertices of the clause subgraph. Then we have

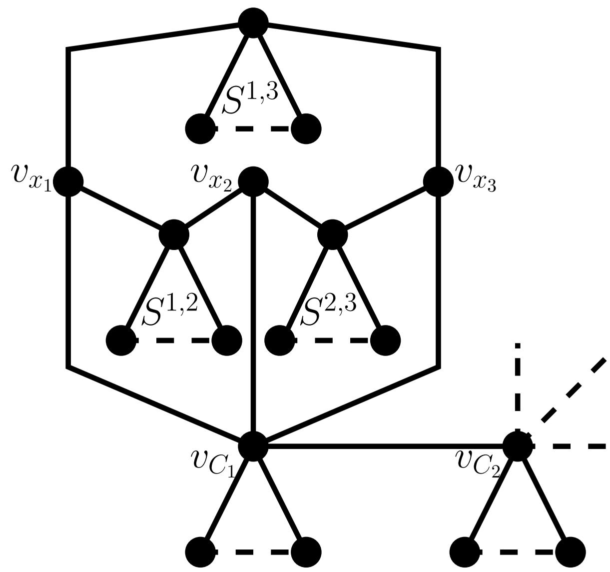

Fig. 2 Connection between the base and clause subgraphs of G_F for a clause $C_1 = (x_1 \vee x_2 \vee x_3)$ of F

$$n_2 \leq 2n + n(n+1) + 3m(n+1) + n + 1$$

since there are exactly $2n$ literals and n pairs of literals of the form $\{\ell_i, \overline{\ell_i}\}$ in F, all the clauses of F can have distinct literals, and the control star S^c has exactly n vertices of degree 1.

The second subgraph of G_F is the *base subgraph*. With each clause C_i in F we associate a *clause vertex* v_{C_i} in the base subgraph that is linked to $n_2 - n$ vertices of degree 1. For each $i \in \{1, ..., m-1\}$, we finally add the edge $v_{C_i}v_{C_{i+1}}$ to $E(G_F)$ so that the clause vertices induce a path in G_F. If we denote by n_1 the number of vertices of the base subgraph of G_F, then we have

$$n_1 = m(n_2 - n + 1).$$

We end up the construction of G_F by adding some edges between the base and clause subgraphs of G_F: for each clause $C_i = (\ell_{i_1} \vee \ell_{i_2} \vee \ell_{i_3})$ in F, we add $v_{C_i}v_{\ell_{i_1}}$, $v_{C_i}v_{\ell_{i_2}}$ and $v_{C_i}v_{\ell_{i_3}}$ to $E(G_F)$. See Figure 2 for an illustration of this connection.

The number of vertices of G_F is $n_1 + n_2$. Thus, the construction of G_F is performed in polynomial time regarding the size of F. Consider now the sequence $\tau_F = (n_1 + n, n_2 - n)$. Since the two elements of τ_F are strictly greater than 1, any part U from a realization R of τ_F in G_F that covers the root vertex of any star subgraph in G_F must also contain all the vertices of degree 1 attached to it. Indeed, if this were not the case, then the graph $G_F - U$ would contain at least two connected components and, thus, the part of R different from U could not induce a connected subgraph of G_F.

For this reason, observe that, because of all the induced stars S_{n_2-n+1} in the base subgraph of G_F, this subgraph must be covered by the part V_1 with size n_1+n of a realization (V_1,V_2) of τ_F in G_F. Starting from this, we then have to add n additional vertices from the clause subgraph of G_F to V_1. For a similar reason as the one above, we can only pick up some literal vertices of G_F since picking up any other of its vertices would disconnect G_F into too many small components. According to our construction, we cannot also add to V_1 two literal vertices v_{ℓ_i} and v_{ℓ_j} such that ℓ_i and ℓ_j are a variable of F and its negation, or appear in a same clause of F, since otherwise this would once again make the subgraph $G_F - V_1$ disconnected.

We can then deduce a 1-in-3 truth assignment of the variables of F from a realization $R=(V_1,V_2)$ of τ_F in G_F and vice-versa. If R is a correct realization of τ_F in G_F, then there are exactly n literal vertices $v_{\ell_{i_1}},...,v_{\ell_{i_n}}$ from the clause subgraph of G_F that belong to V_1. Since $G_F[V_2]$ is connected, setting the literals $\ell_{i_1},...,\ell_{i_n}$ true makes F evaluated true in a 1-in-3 way since no pair of these literals is a variable of F and its negation or appears in a same clause of F. Conversely, if F is satisfiable in a 1-in-3 way, then let $\phi : \{\ell_1,...,\ell_{2n}\} \rightarrow \{0,1\}$ be a satisfying 1-in-3 truth assignment of its literals. Then observe that (V_1,V_2), where

- V_1 contains all the vertices from the base subgraph of G_F and every literal vertex v_{ℓ_i} from the clause subgraph of G_F such that $\phi(\ell_i)=1$,
- $V_2 = V(G_F) - V_1$,

is a correct realization of τ_F in G_F according to the arguments above. □

We finally explain how to generalize the reduction of Theorem 1 so that we get a reduction from 1-IN-3 SAT to k-REALSEQ for any $k \geq 3$.

Theorem 2 *k-REALSEQ is NP-complete for every $k \geq 2$.*

Proof k-REALSEQ is in NP for every $k \geq 2$ as mentioned in the proof of Theorem 1. The proof that k-REALSEQ is NP-complete for every $k \geq 3$ is based on our reduction from 1-IN-3 SAT to 2-REALSEQ. More precisely, we modify the instance resulting from the reduction, i.e. the graph G_F and the sequence τ_F, in such a way that the arguments given in the proof of Theorem 1 are still correct and not altered by the modifications.

For the sake of this proof, let us introduce the following definition. Given a graph H, a vertex $v \in V(H)$ and an arbitrary integer $a \geq 3$, the *(a,v)-star-augmentation of H* is the graph obtained as follows:

1. consider the union of H and a star S^a with $a-1$ vertices of degree 1,
2. add an edge between v and the root of S^a.

An example of an (a,v)-star-augmentation of a graph is depicted in Figure 3. Let us first show that 3-REALSEQ is NP-complete by reduction from 1-IN-3 SAT before generalizing our arguments. From a 3CNF formula F, we construct a graph G_F and a sequence $\tau_F=(n_1,n_2,n_3)$ admissible for G_F such that F is satisfiable in a 1-in-3 way if and only if τ_F is realizable in G_F.

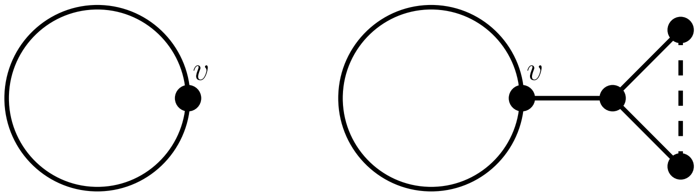

Fig. 3 A graph H and an arbitrary (a, v)-star-augmentation of H

By applying the reduction from 1-in-3 SAT to 2-RealSeq, we get a graph G'_F and a sequence $\tau'_F = (n'_1, n'_2)$ admissible for G'_F that is realizable in G'_F if and only if F admits a 1-in-3 assignment of its variables. Besides, recall that $n'_1, n'_2 \geq 2$. Now consider, as G_F, an (a, v)-star-augmentation of G'_F where $a = n'_1 + n'_2 + 1$ and $v \in V(G'_F)$ is arbitrary, and $\tau_F = (a, n'_1, n'_2)$. In a realization (U, V_1, V_2) of τ_F in G_F, notice that, because $n'_1, n'_2 \geq 2$, the star subgraph S^a of G_F resulting from the star augmentation must be covered entirely by the part U with size a since covering it with one of the other two parts would disconnect G_F into too many small components. Therefore, τ_F is realizable in G_F if and only if τ'_F is realizable in G'_F, and by transitivity we get that F is satisfiable in a 1-in-3 way if and only if τ_F is realizable in G_F.

One can repeat the previous procedure as many times as wanted until τ_F has the requested size. At each step, we get another instance of RealSeq which is equivalent to the previous one but whose sequence has one more element. Said differently, from the instance F of 1-in-3 SAT we first construct an equivalent instance of 2-RealSeq. From this instance of 2-RealSeq is then obtained an equivalent instance of 3-RealSeq thanks to a star-augmentation. With the same construction, we then get an equivalent instance of 4-RealSeq. And so on. All these reduced instances are obtained in polynomial time, and are equivalent to F by transitivity. We thus get that k-RealSeq is NP-complete for every $k \geq 3$. □

3 Complexity of partitioning a graph into connected subgraphs following a prescription

In this section, we investigate the computational complexity of the following decision problem.

Realizable Sequence Under Prescriptions - PrescSeq
Instance: A graph G, a sequence τ, and a prescription P of G.
Question: Is τ realizable in G under P?

Similarly as for RealSeq and its refinement k-RealSeq, we introduce a refined version of PrescSeq obtained by assuming that the sizes of τ and P are fixed.

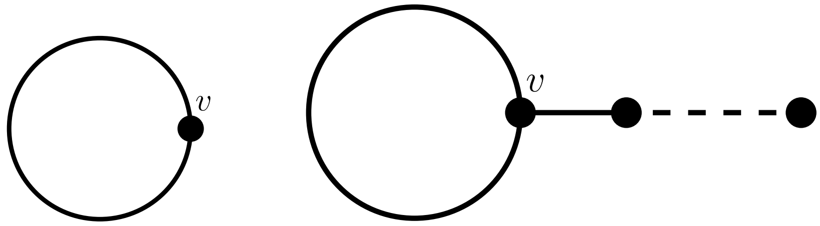

Fig. 4 A graph H and an arbitrary (a, v)-path-augmentation of H

Realizable k-Sequence Under k' Prescriptions - (k, k')-PrescSeq
Instance: A graph G, a k-sequence τ, and a k'-prescription P of G.
Question: Is τ realizable in G under P?

For any problem (k, k')-PrescSeq, we have $k \geq k'$ by definition. Besides, $(k, 0)$-PrescSeq is equivalent to k-RealSeq which was shown to be in P when $k = 1$, and NP-complete for every $k \geq 2$ (Section 2). Note further that the answer to an instance of $(1, 1)$-PrescSeq is *yes* if and only if G is connected. Therefore, $(1, 1)$-PrescSeq is in P.

We now prove that the remaining problems (k, k')-PrescSeq, i.e. with $k \geq 2$ and $1 \leq k' \leq k$, are NP-complete.

Theorem 3 (k, k')-PrescSeq *is NP-complete whenever* $k \geq 2$*.*

Proof One can clearly modify the checking algorithm for RealSeq we gave in the proof of Theorem 1 so that it also takes the prescription P as an input and makes sure that the vertices of P belong to the corresponding parts of R. This modification does not alter the complexity of the algorithm. Therefore, PrescSeq is in NP.

Let k and k' be fixed. Clearly, if $k' = 0$, then (k, k')-PrescSeq is NP-complete by Theorem 2. Suppose thus that $k' \geq 1$. We show that (k, k')-PrescSeq is NP-complete thanks to our reduction from 1-in-3 SAT to k-RealSeq (Theorem 2) and the following construction. Let $a \geq 1$ be an arbitrary positive integer and v be an arbitrary vertex of some graph H. The (a, v)*-path-augmentation of* H is the graph obtained from H as follows:

1. consider the union of H and P_a, a path of order a whose vertices are consecutively denoted by $u_1, ..., u_a$;
2. add an edge between u_1 and v.

This construction is depicted in Figure 4. First suppose that $k - k' \geq 2$. From F, start by constructing a graph G_F and a sequence $\tau_F = (n_1, ..., n_{k-k'})$ with size $k - k'$ admissible for G_F such that F is satisfiable in a 1-in-3 way if and only if τ_F is realizable in G_F. This graph G_F and sequence τ_F may be obtained thanks to the reduction from Theorem 2 since $k - k' \geq 2$. Let us now denote by G'_F the graph obtained from G_F by performing k' arbitrary path-augmentations, e.g. one (a_1, v)-path-augmentation, one (a_2, v)-path-augmentation, etc., for some $v \in V(G_F)$ and integers $a_1, ..., a_{k'} \geq 1$. Let

$u_1, ..., u_{k'}$ denote the vertices with degree 1 of the resulting hanging paths, where u_i is the endvertex of the i^{th} path-augmentation. Finally, let $\tau'_F = (a_1, ..., a_{k'}, n_1, ..., n_{k-k'})$ and $P'_F = (u_1, ..., u_{k'})$ be a k-sequence admissible for G'_F and a k'-prescription of G'_F, respectively.

Since the first k' parts $U_1, ..., U_{k'}$ of a realization of τ'_F in G'_F under P'_F must induce connected subgraphs of G'_F on $a_1, ..., a_{k'}$ vertices, respectively, including $u_1, ..., u_{k'}$, respectively, the only way for choosing the part U_i is to pick up every vertex resulting from the i^{th} path-augmentation. Once these parts have been picked up, we still have to find a realization $(V_1, ..., V_{k-k'})$ of the remaining sequence $(n_1, ..., n_{k-k'}) = \tau_F$ in the remaining graph $G'_F - \bigcup_{i=1}^{k'} U_i = G_F$. Hence, τ'_F is realizable in G'_F under P'_F if and only if τ_F is realizable in G_F. By transitivity, we get that F is satisfiable in a 1-in-3 way if and only if τ'_F is realizable in G'_F under P'_F. This reduction can clearly be performed in polynomial time.

Note that this reduction does not work when $k - k' \in \{0, 1\}$ since, in this situation, too much prescribed vertices are requested. But recall that, in the reduction from 1-in-3 SAT to 2-RealSeq, some vertices from the base and clause subgraphs of G_F, respectively, have to be covered by the parts with size n_1 and n_2, respectively, of a realization of τ_F in G_F. Thus, we could request up to 2 prescriptions, and directly get that $(2, 1)$- and $(2, 2)$-PrescSeq are NP-complete. By performing the same reduction scheme as above but from one of these two problems, we get that (k, k')-PrescSeq is also NP-complete when $k - k' \in \{0, 1\}$. □

4 Complexity of partitioning a graph with given connectivity into connected subgraphs following a prescription

In the introduction section, we mentioned a famous result proved independently by Györi and Lovász on the problem of realizing sequences in q-connected graphs. Using our terminology, this result may be formulated as follows [6, 7].

Theorem 4 (Györi and Lovász, independently) *Every sequence with size $k \leq q$ admissible for a q-connected graph G is realizable in G under k prescriptions.*

Theorem 4 implies that the answer to every instance of (k, k)-PrescSeq such that G is a q-connected graph and τ is admissible for G is *yes* whenever $k \leq q$. We now show that this easiness result is in some sense tight, i.e. that prescribing strictly more than q vertices while partitioning a q-connected graph is difficult in general.

Theorem 5 (k, k')-PrescSeq *is NP-complete when restricted to q-connected graphs for every $q \geq 1$ whenever $q < k' \leq k$.*

Proof The NP part of the claim derives from the NP part of PrescSeq which was shown in the proof of Theorem 3. The hardness part can be proved thanks to our previous reductions from 1-in-3 SAT.

First, because (k, k')-PrescSeq is NP-complete for every $k \geq 2$ and our proof of this statement was obtained by reducing instances of 1-in-3 SAT to 1-connected graphs (see Theorem 3), the statement holds for $q = 1$.

We now prove the general case, i.e. $q \geq 2$. Let $k' > q$ and $k \geq k'$ be fixed. Start from a 3CNF formula F, and produce a graph G_F, a sequence $\tau_F = (n_1, ..., n_{k-q+1})$ admissible for G_F, and a $(k'-q+1)$-prescription $(u_1, ..., u_{k'-q+1})$ P_F of G_F such that F is 1-in-3 satisfiable if and only if τ_F is realizable in G_F under P_F. This reduced instance may be obtained thanks to the reduction given in the proof of Theorem 1, and the star- and path-augmentation constructions introduced in the proofs of Theorems 2 and 3, respectively. Now consider the following instance of (k, k')-PrescSeq.

- G'_F is obtained by successively adding $q-1$ universal vertices $v_1, ..., v_{q-1}$ to G_F, i.e. vertices joined to all other vertices of the graph.
- $\tau'_F = (1, ..., 1, n_1, ..., n_{k-q+1})$ is a sequence admissible for G'_F with $q-1$ 1's.
- $P'_F = (v_1, ..., v_{q-1}, u_1, ..., u_{k'-q+1})$ is a prescription of G'_F.

Clearly, G'_F is q-connected since G_F is 1-connected, and τ'_F and P'_F have size k and k', respectively. Besides, since prescribing a vertex to a part with size 1 is like removing it from the graph, what is left once the vertices $v_1, ..., v_{q-1}$ have been assigned to parts with size 1 of a realization is G_F, τ_F and P_F. Therefore, τ'_F is realizable in G'_F under P'_F if and only if τ_F is realizable in G_F under P_F. By transitivity, we get that F is 1-in-3 satisfiable if and only if τ'_F is realizable in G'_F under P'_F. □

5 Some Π_2^p problems

In this section, we investigate the relationship between some graph partition problems derived from our definitions and the Π_2^p complexity class. We start with the following problem.

AP Graph
Instance: A graph G.
Question: Is G an AP graph?

This problem is not known to belong to either NP or co-NP. However, it is clearly in Π_2^p since one can design a polynomial-time algorithm that takes G and a sequence τ admissible for G as input and checks that τ is not realizable in G using an oracle for RealSeq.

Consider next the following problem.

AP+k Graph
Instance: A graph G.
Question: Is G an AP+k graph?

Clearly, AP+k Graph is also in Π_2^p for every $k \geq 1$. Indeed, recall that PrescSeq is NP-complete whatever is the number of prescribed vertices (Theorem 3). One can thus design a similar algorithm as the one we just mentioned for AP Graph, except that this algorithm uses an oracle for PrescSeq.

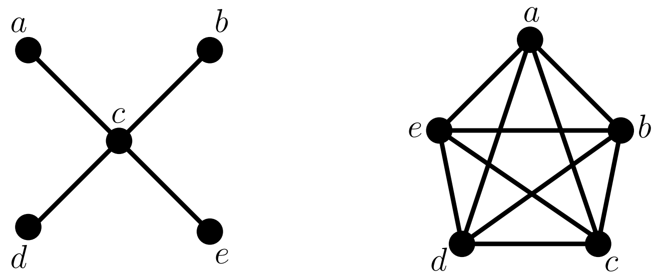

Fig. 5 The graphs $K_{1,4}$ and K_5

We do not know whether AP Graph and AP+k Graph are Π_2^p-complete problems. Indeed, to design a polynomial-time reduction from a Π_2^p-complete problem A to one of these two problems, it would be necessary to "translate" the restrictions associated with an instance of A to some graph substructures just like we did in the proof of Theorem 1 by introducing a lot of star subgraphs in the reduced graphs. But introducing these graph substructures generally makes the whole graph being not AP. That is why, for example, our reduction from 1-in-3 SAT to RealSeq does not seem to be generalizable into some reduction from a Π_2^p-complete version of 1-in-3 SAT to AP Graph.

Most of Π_2^p problems are of the form *"For every X, is there a Y such that...?"*. The AP Graph and AP+k Graph problems clearly catch this form since one could reformulate them as *"For every admissible sequence, is there a realization such that...?"*. However, in most of Π_2^p-complete problems, the two input objects X and Y have the same nature (e.g. truth assignments, sets of vertices, etc.) while it is not the case for AP Graph and AP+k Graph. This is another reason why it seems difficult to design a reduction from one classical Π_2^p-complete problem to one of these two problems.

In order to show that graph partition problems are not "incompatible" with the notion of Π_2^p-complete problems, we introduce another problem. Let G be a graph and $\tau = (n_1, ..., n_p)$ be a sequence admissible for G. Given a $\ell \in \{1, ..., p\}$, a *n_ℓ-partition-level for τ and G* is a set L_ℓ of subsets of $V(G)$ that induce connected subgraphs of G with order n_ℓ. A *$(n_1, ..., n_\ell)$-partition-hierarchy L for τ and G* is a collection $L = (L_1, ..., L_\ell)$ of n_i-partition-levels for τ and G for i up to ℓ such that no subsets in L_i and L_j intersect for $i \neq j$. We finally say that τ is *realizable in G under L* if for every collection of subsets $(V_1, ..., V_\ell)$ from the partition levels of L such that $V_1 \in L_1, ..., V_\ell \in L_\ell$ there exists a realization $(V_1, ..., V_\ell, ..., V_p)$ of τ in G. In other words, we are given partial realizations of τ in G, i.e. some ways for picking up the parts associated with the ℓ first elements of τ, whose parts are dispatched into ℓ partition levels, and we ask whether each of these partial realizations is extendable to a whole realization of τ in G. A partition hierarchy is actually a compact way to describe a large number of partial realizations.

As an illustration of these definitions, consider the two graphs $K_{1,4}$ and K_5 of Figure 5. Let $\tau = (1, 1, 3)$ be a sequence admissible for $K_{1,4}$ and K_5, let $L_1 = (\{a\}, \{c\})$ and $L_2 = (\{b\}, \{e\})$ be two 1-partition-levels for τ and both $K_{1,4}$ and K_5, and $L = (L_1, L_2)$ be a $(1,1)$-partition-hierarchy for τ and both $K_{1,4}$ and K_5. Clearly, τ is not realizable in $K_{1,4}$ under L since $(\{c\}, \{b\}, V(K_{1,4}) - \{c, b\})$ is not a correct realization of τ in $K_{1,4}$. However, τ is realizable in K_5 under L since $(\{a\}, \{b\}, V(K_5) - \{a, b\})$, $(\{a\}, \{e\}, V(K_5) - \{a, e\})$, $(\{c\}, \{b\}, V(K_5) - \{c, b\})$ and $(\{c\}, \{e\}, V(K_5) - \{c, e\})$ are correct realizations of τ in K_5.

We now investigate the computational complexity of the problem associated with the definition above.

Dynamic Realizable Sequence - DynRealSeq
Instance: A graph G, a sequence $\tau = (n_1, ..., n_{p'}, ..., n_p)$ with size $p \geq p'$, and a $(n_1, ..., n_{p'})$-partition-hierarchy L for τ and G.
Question: Is τ realizable in G under L?

As understood above, DynRealSeq is a Π_2^p-complete problem. Our proof of this claim is based on our reduction from 1-in-3 SAT to RealSeq (Section 2). In order to reuse it, we need a Π_2^p-complete version of 1-in-3 SAT.

∀∃1-in-3 SAT
Instance: A 3CNF formula F over variables $X \cup Y$ and clauses $\{C_1, ..., C_m\}$, where $X = \{x_1, ..., x_{n'}\}$, $Y = \{x_{n'+1}, ..., x_n\}$ and $n' \leq n$.
Question: For every truth assignment of the variables of X, does there exist a truth assignment of the variables of Y such that F is satisfied in a 1-in-3 way?

We first show below that ∀∃1-in-3 SAT is Π_2^p-complete by reduction from the following classical Π_2^p-complete problem.

∀∃3SAT
Instance: A 3CNF formula F over variables $X \cup Y$, where $X = \{x_1, ..., x_{n'}\}$, $Y = \{x_{n'+1}, ..., x_n\}$ and $n' \leq n$, and clauses $\{C_1, ..., C_m\}$.
Question: For every truth assignment of the variables of X, does there exist a truth assignment of the variables of Y such that F is satisfied?

Lemma 6 ∀∃1-in-3 SAT *is Π_2^p-complete.*

Proof ∀∃1-in-3 SAT is clearly in Π_2^p. One can indeed design an algorithm that takes F and a truth assignment ϕ_1 to the variables of X for which there is no truth assignment ϕ_2 to the variables in Y making F evaluated in a 1-in-3 way as input. It just has to check that ϕ_2 does not exist thanks to an oracle dealing with 1-in-3 SAT. Such a checking algorithm runs in polynomial time regarding the size of F.

We now show that ∀∃1-in-3 SAT is Π_2^p-complete by reduction from ∀∃3SAT. From a 3CNF formula F over variables $X \cup Y$, we construct a new 3CNF formula F' over variables $X' \cup Y'$ such that for every truth assignment ϕ_1 to the variables in X there exists a truth assignment ϕ_2 to the variables in Y making F evaluated true if and only if for every truth assignment ϕ'_1 to the variables

$(\phi_3(\ell_{i_1}), \phi_3(\ell_{i_2}), \phi_3(\ell_{i_3}))$	$\phi'_2(a_i)$	$\phi'_2(b_i)$	$\phi'_2(c_i)$	$\phi'_2(d_i)$	$\phi'_2(e_i)$	$\phi'_2(f_i)$
$(1,0,0)$	1	0	0	0	0	0
$(0,1,0)$	0	0	1	0	0	0
$(0,0,1)$	0	0	0	0	1	0
$(1,1,0)$	1	0	0	1	0	0
$(1,0,1)$	1	0	0	0	0	1
$(0,1,1)$	0	0	1	0	0	1
$(1,1,1)$	1	0	0	1	0	1

Table 1 Truth assignment of ϕ'_2 for the variables in $Y - Y'$

in X' there exists a truth assignment ϕ'_2 to the variables in Y' such that F' is evaluated true in a 1-in-3 way.

The reduction is straightforward. First, replace each clause $C_i = (\ell_{i_1} \vee \ell_{i_2} \vee \ell_{i_3})$ in F by four clauses $(\overline{\ell_{i_1}} \vee a_i \vee b_i)$, $(\overline{\ell_{i_2}} \vee c_i \vee d_i)$, $(\overline{\ell_{i_3}} \vee e_i \vee f_i)$ and $(a_i \vee c_i \vee e_i)$ in F' where a_i, b_i, c_i, d_i, e_i and f_i are six new variables associated with C_i. Finally, let $X' = X$ and $Y' = Y \cup \bigcup_{i=1}^{m} \{a_i, b_i, c_i, d_i, e_i, f_i\}$. Note that F' has $4m$ clauses and may be obtained easily.

First suppose that for every truth assignment ϕ'_1 to the variables of X' there exists a truth assignment ϕ'_2 to the variables in Y' such that F' is satisfied in a 1-in-3 way. Because every clause of F' has exactly one true literal under ϕ'_1 and ϕ'_2, it means that only one element in $\{a_i, c_i, e_i\}$ is evaluated true by ϕ'_2 for every $i \in \{1, ..., m\}$. Let us suppose that for such an i we have $\phi'_2(a_i) = 1$ and $\phi'_2(c_i) = \phi'_2(e_i) = 0$ without loss of generality. Thus, we have ℓ_{i_1} evaluated true by either ϕ'_1 or ϕ'_2. It follows that the following truth assignment ϕ_1 and ϕ_2 of the variables in X and Y, respectively,

- $\phi_1 = \phi'_1$,
- $\phi_2(x) = \phi'_2(x)$ for every $x \in Y$,

is such that F is satisfied. Conversely, suppose that for every truth assignment ϕ_1 to the variables in X there is a truth assignment ϕ_2 to the variables in Y such that F has all its clauses satisfied under ϕ_1 and ϕ_2. We explain how to get a truth assignment ϕ'_2 to the variables in Y' so that F' is evaluated true in a 1-in-3 way under ϕ'_2 and the truth assignment $\phi'_1 = \phi_1$ to the variables in X'. First, let $\phi'_2(x) = \phi_2(x)$ for every $x \in Y$. We then have to provide a truth assignment of a_i, b_i, c_i, d_i, e_i and f_i via ϕ'_2 for every $i \in \{1, ..., m\}$. This assignment depends on the number of true literals in $C_i = (\ell_{i_1} \vee \ell_{i_2} \vee \ell_{i_3})$ via ϕ_1 and ϕ_2. Let $\phi_3 : X \cup Y \to \{0, 1\}$ be the truth assignment of the variables in $X \cup Y$ deduced from ϕ_1 and ϕ_2 as follows:

- if $i \in \{1, ..., n'\}$, then $\phi_3(x_i) = \phi_1(x_i)$;
- if $i \in \{n' + 1, ..., n\}$, then $\phi_3(x_i) = \phi_2(x_i)$.

Consider now that the images of the a_i's, b_i's, c_i's, d_i's, e_i's and f_i's by ϕ'_2 are the ones depicted in Table 1. It should then be clear that F' is evaluated true in a 1-in-3 way under ϕ'_1 and ϕ'_2.

□

We finally prove that DynRealSeq is Π_2^p-complete.

Theorem 7 DynRealSeq *is* Π_2^p*-complete.*

Proof DynRealSeq is clearly a Π_2^p problem. One can provide a combination of parts $(V_1, ..., V_{p'})$ from the $(n_1, ..., n_{p'})$-partition-hierarchy for τ and G of the problem instance to a polynomial-time algorithm checking that these parts cannot be extended to a realization of τ in G. It just has to make sure that τ is admissible for G, and the sequence $(n_{p'+1}, ..., n_p)$ is not realizable in $G - \bigcup_{i=1}^{p'} V_i$ using an oracle for RealSeq. Note further that DynRealSeq can be neither in co-NP (for the same reason as RealSeq is not in co-NP) nor in NP since the number of partial realizations encoded by the partition-hierarchy may be exponential regarding the input size.

We now show that DynRealSeq is complete in Π_2^p by reduction from ∀∃1-in-3 SAT (Π_2^p-complete by Lemma 6). Our reduction is inspired by the reduction from 1-in-3 SAT to RealSeq we gave in the proof of Theorem 1. Remember that in this reduction, setting a variable of F to true is simulated in an instance of RealSeq by adding a literal vertex of G_F to the part with size $n_1 + n$ of a realization of τ_F in G_F. We here somehow want to keep that relationship between setting a variable of F to true and putting a literal vertex of the G_F into a part of the realization. Given a truth assignment ϕ_1 to the variables in X, it means that we have to check whether every partial realization of τ_F in G_F whose part with size $n_1 + n$ contains the literal vertices associated with the true literals via ϕ_1 is extendible to a realization of τ_F in G_F. All these possible partial realizations are considered using a partition-hierarchy for τ_F and G_F.

First of all, let G_F be the graph obtained from F using the reduction we gave in the proof of Theorem 1. Then, let $\tau_F = (1, ..., 1, n_1 + n - n', n_2 - n)$ be a sequence with size $n' + 2$ admissible for G_F, let $L_i = \{\{v_{x_i}\}, \{v_{\overline{x_i}}\}\}$ be a 1-partition-level for τ_F and G_F for every $x_i \in X$, and $L = \bigcup_{i=1}^{n'} L_i$ be a $(1, ..., 1)$-partition-hierarchy for τ_F and G_F. With a truth assignment ϕ_1 of the variables in X setting n' literals of F to true is then associated a combination of vertex-disjoint subsets $(V_1, ..., V_{n'})$ from the 1-partition-levels in L, where $V_i = \{x_i\}$ if $\phi_1(x_i) = 1$ or $V_i = \{\overline{x_i}\}$ otherwise.

Let us now suppose that for every truth assignment ϕ_1 to the variables in X there exists a truth assignment ϕ_2 to the variables of Y such that F is evaluated true in a 1-in-3 way. Then the realization $(V_1, ..., V_{n'+2})$ of τ_F in G_F, where

- for every $i \in \{1, ..., n'\}$, we have $V_i = \{v_{x_i}\}$ if $\phi_1(x_i) = 1$ or $V_i = \{v_{\overline{x_i}}\}$ otherwise,
- $V_{n'+1}$ contains all the vertices from the base subgraph of G_F and every literal vertex v_{ℓ_i} of the clause subgraph of G_F such that $\phi_2(\ell_i) = 1$,
- $V_{n'+2} = V(G_F) - \bigcup_{i=1}^{n'+1} V_i$,

is correct according to the arguments we gave in the proof of Theorem 1. Conversely, suppose that every combination $(V_1, ..., V_{n'})$ of subsets from the

1-partition-levels of L is extendable to a realization $(V_1, ..., V_{n'+2})$ of τ_F in G_F. As explained before, the partition $(V_1, ..., V_{n'})$ is associated with a truth assignment ϕ_1 to the variables in X, and from the literal vertices contained in $V_{n'+1}$ we can deduce a truth assignment ϕ_2 to the variables in Y (see the proof of Theorem 1). Clearly, F is evaluated true in a 1-in-3 way under ϕ_1 and ϕ_2. □

References

1. D. Barth, O. Baudon, and J. Puech. Decomposable trees: a polynomial algorithm for tripodes. *Discret. Appl. Math.*, 119(3):205–216, July 2002.
2. D. Barth and H. Fournier. A degree bound on decomposable trees. *Discret. Math.*, 306(5):469–477, 2006.
3. O. Baudon, J. Bensmail, J. Przybyło, and M. Woźniak. Partitioning powers of traceable or Hamiltonian graphs. *Preprint*, 2012. Available at http://hal.archives-ouvertes.fr/hal-00687278.
4. M.E. Dyer and A.M. Frieze. On the complexity of partitioning graphs into connected subgraphs. *Discret. Appl. Math.*, 10:139–153, 1985.
5. N. Guttmann-Beck and R. Hassin. Approximation algorithms for min-max tree partition. *J. Algorithms.*, 24(2):266–286, 1997.
6. E. Györi. On division of graphs to connected subgraphs. In *Combinatorics*, pages 485–494, Colloq. Math. Soc. János Bolyai 18, 1978.
7. L. Lovász. A homology theory for spanning trees of a graph. *Acta Math. Acad. Sci. Hung.*, 30(3-4):241–251, 1977.