UNIVERSITE PARIS DAUPHINE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision
CNRS UMR 7243

CAHIER DU LAMSADE

311

Juin 2011

Reoptimization of maximum weight induced hereditary subgraph problems

Nicolas Boria, Jérôme Monnot, Vangelis Th. Paschos

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

Reoptimization of maximum weight induced hereditary subgraph problems*

Nicolas Boria Jérôme Monnot Vangelis Th. Paschos§

LAMSADE, CNRS UMR 7243 and Université Paris-Dauphine

{boria,monnot,paschos}@lamsade.dauphine.fr

§Institut Universitaire de France

June 15, 2011

Abstract

The reoptimization issue studied in this paper can be described as follows: given an instance I of some problem Π, an optimal solution OPT for Π in I and an instance I' resulting from a local perturbation of I that consists of insertions or removals of a small number of data, we wish to use OPT in order to solve Π in I', either optimally or by guaranteeing an approximation ratio better than that guaranteed by an ex nihilo computation and with running time better that that needed for such a computation. We use this setting in order to study weighted versions of several representatives of a broad class of problems known in the literature as maximum induced hereditary subgraph problems. The main problems studied are MAX INDEPENDENT SET, MAX k-COLORABLE SUBGRAPH, MAX P_k-FREE SUBGRAPH, MAX SPLIT SUBGRAPH and MAX PLANAR SUBGRAPH. We also show, how the techniques presented allow us to handle also BIN PACKING.

1 Introduction

Hereditary problems in graphs, also known as maximal subgraph problems, include a wide range of classical combinatorial optimization problems, such as MAX INDEPENDENT SET or MAX H-FREE SUBGRAPH. Most of these problems are known to be **NP**-hard, and even inapproximable within any constant approximation ratio unless **P** = **NP** [24, 31]. However, some of them, and in particular MAX INDEPENDENT SET, have been intensively studied in the polynomial approximation framework both in the general case [13, 20], as well as in particulars classes of graphs where some constant approximation ratios can be achieved in polynomial time [5, 6, 17, 21, 23, 25].

Also, some of these problems have been studied in dynamic settings, where instances are allowed to evolve over time, and solutions must be provided after each instance modification. In particular, a whole class of hereditary problems has been studied in the online setting [16], and MAX INDEPENDENT SET has been analyzed in both the online [19, 22], and a priori optimization frameworks [26].

In what follows, we present simple approximation algorithms and inapproximability bounds for various hereditary problems in the reoptimization setting, which can be described as follows: considering an instance I of a given problem Π with a known optimum OPT, and an instance I' which results from a local perturbation of I, can the information provided by OPT be used to

*Research supported by the French Agency for Research under the DEFIS program TODO, ANR-09-EMER-010

solve I' in a more efficient way (i.e., with a lower complexity and/or with a better approximation ratio) than if this information wasn't available?

The reoptimization setting was introduced in [1] for METRIC TSP. Since then, many other optimization problems were discussed in this setting, including STEINER TREE [7, 10, 11, 18], MINIMUM SPANNING TREE [14], as well as various versions of TSP [4, 9, 12]. In all cases, the goal is to propose reoptimization algorithm that outperform their deterministic counterparts in terms of complexity and/or approximation ratio. In [8], the MAX INDEPENDENT SET problem, as well as MIN VERTEX COVER and MIN SET COVER problems, are discussed in a similar setting up to the fact that perturbations there concerned the edge-set of the initial graph. The authors of [8] manage to provide optimal approximation results under the basic assumption that the initial solution is not necessarily optimal but ρ-approximate.

When one deals with hereditary problems, and I' results from a perturbation of the vertex set (insertion or deletion), solutions of I remain feasible in I'. This property is very interesting when reoptimizing hereditary problems, and makes most of them APX in the reoptimization setting. For exemple, a very simple algorithm provides a (1/2)-approximation for a whole class of hereditary problems when a single vertex is inserted [3]. In what follows, we improve on this result by presenting algorithms designed for four specific hereditary problems, and also provide inapproximability bounds. We also discuss the reoptimization setting where vertices are deleted, which, as we will see, is much harder to approximate.

The paper is organized as follows: general properties regarding hereditary problems are presented in Section 2, while Sections 3 and 4 present approximation and inapproximability results regarding respectively vertex insertion and deletion. Finally, in Section 5 the BIN PACKING problem is studied.

2 Preliminaries

Before presenting properties and results regarding reoptimization problems, we will first give formal definitions of what are reoptimization problems, reoptimization instances, and approximate reoptimization algorithms:

Definition 1. An optimization problem Π is given by a quadruple $(\mathcal{I}_\Pi, Sol_\Pi, m_\Pi, goal(\Pi))$ where:

- $\mathcal{I}_\Pi$ is the set of instances of Π;
- given $I \in \mathcal{I}_\Pi$, $Sol_\Pi(I)$ is the set of feasible solutions of I;
- given $I \in \mathcal{I}_\Pi$, and $S \in Sol_\Pi(I)$, $m_\Pi(I, S)$ denotes the value of the solution S of the instance I, m_Π is called the objective function;
- $goal(\Pi) \in \{\min, \max\}$. ∎

A reoptimization problem $R\Pi$ is given by a pair $(\Pi, R_{R\Pi})$ where:

- Π is an optimization problem as defined in Definition 1;
- $R_{R\Pi}$ is a rule of modification on instances of Π, such as addition, deletion or alteration of a given amount of data; given $I \in \mathcal{I}_\Pi$ and $R_{R\Pi}$, $modif_{R\Pi}(I, R_{R\Pi})$ denotes the set of instances resulting from applying modification $R_{R\Pi}$ to I; notice that $modif_{R\Pi}(I, R_{R\Pi}) \subset \mathcal{I}_\Pi$.

For a given reoptimization problem $R\Pi(\Pi, R_{R\Pi})$, a reoptimization instance $I_{R\Pi}$ of $R\Pi$ is given by a triplet (I, S, I'), where:

- I denotes an instance of Π, referred to as the *initial* instance;
- S denotes a feasible solution for Π on the initial instance I;
- I' denotes an instance of Π in $modif_{R\Pi}(I, R_{R\Pi})$; I' is referred to as the *perturbed* instance.

For a given instance $I_{R\Pi}(I, S, I')$ of $R\Pi$, the set of feasible solutions is $Sol_\Pi(I')$.

Definition 2. For a given optimization problem $R\Pi(\Pi, R_{R\Pi})$, a reoptimization algorithm `A` is said to be a ρ-approximation reoptimization algorithm for $R\Pi$ if and only if:

- `A` returns a feasible solution on all instances $I_{R\Pi}(I, S, I')$;
- `A` returns a ρ-approximate solution on all reoptimization instances $I_{R\Pi}(I, S, I')$ where S is an optimal solution for I. ∎

Note that Definition 2 is the most classical definition found in the literature, as well as the one used in this paper. However, an alternate (and more restrictive) definition exists (used for example in [7, 10, 11]), where a ρ_1-approximation reoptimization algorithm for $R\Pi$ is supposed to ensure a $\rho_1\rho_2$ approximation on any reoptimization instance $I_{R\Pi}(I, S, I')$ where S is a ρ_2 approximate solution in the initial instance I.

A property $\mathcal{P}$ on a graph is hereditary if the following holds: if the graph satisfies $\mathcal{P}$, then $\mathcal{P}$ is also satisfied by all its induced subgraphs. Following this definition, independence, planarity, bipartiteness are three examples of hereditary properties: in a given graph, any subset of an independent set is an independent set itself, and the same holds for planar and bipartite subgraphs. On the opposite hand, connectivity is no hereditary property since there might exist some subsets of G whose removal disconnect the graph. It is also well known that any hereditary property in graphs can be characterized by a set of forbidden subgraphs or minors [29].

In other words a property $\mathcal{P}$ is hereditary if and only if, there is a set of graphs H such that every graph that verifies $\mathcal{P}$ does not admit any graph in H as a minor or as an induced subgraph. To revisit the three examples of hereditary properties presented before:

- an independent set is characterized by one forbidden subgraph: a K_2 (a clique on 2 vertices, i.e., an edge).
- a planar graph is characterized by two forbidden minors: K_5 (a clique on 5 vertices), and $K_{3,3}$ (a complete bipartite subgraph with both its color-classes of size 3). This result is known as Wagner's Theorem [30].
- a bipartite graph is characterized by a infinite set of forbidden subgraphs: all odd cycles $H = \{C_{2n+1}, n \geqslant 1\}$.

Definition 3. Let $G(V, E, w)$ be a vertex-weighted graph with $w(v) \geqslant 0$, for any $v \in V$. The max induced subgraph with property $\mathcal{P}$ problem (or, for short, maximum subgraph problem) is the problem consisting, given a graph $G(V, E)$, of finding a subset of vertices S such that $G[S]$ satisfies a given property $\mathcal{P}$ and that maximizes $w(S) = \sum_{v \in S} w(v)$. We call hereditary problems all such problems where $\mathcal{P}$ is a hereditary property. ∎

For instance, MAX WEIGHTED INDEPENDENT SET, MAX WEIGHTED INDUCED BIPARTITE SUBGRAPH, MAX WEIGHTED INDUCED PLANAR SUBGRAPH are three classical hereditary problems that correspond to the three hereditary properties as defined in Definition 3.

As it is proved in [24] (see Theorem 1 just below) most hereditary problems are highly inapproximable unless $\mathbf{P} = \mathbf{NP}$.

Theorem 1. ([24]) *There exists an $\varepsilon \in (0,1)$ such that the maximum subgraph problem cannot be approximated with ratio $n^{-\varepsilon}$ in polynomial time for any nontrivial hereditary property that is false for some clique or independent set, or more generally is false for some complete multipartite graph, unless* $\mathbf{P} = \mathbf{NP}$.

Throughout the paper, all inapproximability results will be obtained by the same technique, which we sketch out here.

Considering an unweighted graph $H(V, E)$ on which one wants to solve a given hereditary problem Π, known to be inapproximable within any constant ratio, we build a reoptimization instance I_p, where p denotes a vector of fixed size (i.e., independent of the size n of G; so, $|p|$ is a fixed constant) that contains integer parameters between 1 and n. This instance is characterized by an initial graph G_p (that contains H), with a known solution, and a perturbed instance G'_p.

Then, we prove that, for some specific (yet unknown) value p' of the parameter vector p, an optimal solution can be easily determined in the initial graph $G_{p'}$, and a ρ-approximate solution $S_{p'}$ in $G'_{p'}$ necessarily induces a solution $S_{p'}[V]$ in H, that is a constant approximation for the initial problem.

Considering that the vector p can take at most $n^{|p|}$ possible values, it is possible in polynomial time to build all instances I_p, to run the polynomial ρ-approximation algorithm on all of them, and to return the best set $S_{p^*}[V]$ as solution for Π in H. The whole procedure is polynomial and ensures a constant-approximation for Π, which is impossible unless $\mathbf{P} = \mathbf{NP}$, so that a ρ-approximation algorithm cannot exist for the considered reoptimization version of Π, unless $\mathbf{P} = \mathbf{NP}$.

In the sequel, G_p and G'_p will denote initial and perturbed instances, while OPT_p and OPT'_p will denote optimal solutions in G_p and G'_p, respectively. For simplicity and when no confusion arises, we will omit subscript p. The function w refers to the weight function, taking a vertex, a vertex set, or a graph as input (the weight of a graph is defined as the sum of weights of its vertices). Finally, note that throughout the whole paper, the term "subgraph" will always implicitly refer to "induced subgraph".

3 Vertex insertion

Under vertex insertion, the inapproximability bounds of Theorems 1 and are easily broken. In [3], a very simple strategy, denoted by `R1` in what follows, provides a (1/2)-approximation for any hereditary problem. This strategy consists of outputting the best solution among the newly inserted vertex and the initial optimum. Moreover, this strategy can also be applied when a constant number h of vertices is inserted: it suffices to output the best solution between an optimum in the h newly inserted vertices (that can be found in $O(2^h)$ through exhaustive search) and the initial optimum. The 1/2 approximation ratio is also ensured in this case [3].

Note that an algorithm similar to `R1` was proposed for KNAPSACK in [2]. Indeed, this problem, although not being a graph problem, it is hereditary in the sense defined above, so that returning the best solution between a newly inserted item and the initial optimum ensures a (1/2)-approximation ratio. The authors also show that any reoptimization algorithm that does not consider objects discarded by the initial optimal solution cannot have ratio better than 1/2.

In the following, we start by proving that this approximation ratio is the best constant approximation ratio one can achieve for the MAX INDEPENDENT SET problem (Section 3.1), unless $\mathbf{P} = \mathbf{NP}$. Then, we present other simple polynomial constant-approximation strategies, as well as inapproximability bounds for various hereditary problems: MAX k-COLORABLE SUBGRAPH (Section 3.2), MAX P_k-FREE SUBGRAPH (Section 3.3), MAX SPLIT SUBGRAPH (Section 3.4), and MAX PLANAR SUBGRAPH (Section 3.5).

3.1 MAX INDEPENDENT SET

Since MAX INDEPENDENT SET is a hereditary problem, strategy `R1` provides a simple and fast (1/2)-approximation in the reoptimization setting under insertion of one vertex. We will now prove that this ratio is the best one can hope, unless $\mathbf{P} = \mathbf{NP}$.

Proposition 1. *In the reoptimization setting, under one vertex insertion,* MAX INDEPENDENT SET *is inapproximable within ratio* $1/2 + \varepsilon$ *in polynomial time, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. By contradiction, assume that there exists a reoptimization approximation algorithm `A` for MAX INDEPENDENT SET, which, in polynomial time, computes a solution with approximation ratio bounded by $1/2 + \varepsilon$. Now, consider a graph $H(V, E)$. All n vertices in V have weight 1, and no assumption is made on V. Note that in such a graph (which is actually unweighted), MAX INDEPENDENT SET is inapproximable within any constant ratio, unless $\mathbf{P} = \mathbf{NP}$.

We will now make use of `A` to build an ε-approximation for MAX INDEPENDENT SET in H, and thus prove that such an algorithm cannot exist. Denote by α the independence number associated with H, that is, the - unknown - cardinality of an optimal independent set in H, and consider the following instance I_α of MAX INDEPENDENT SET in the reoptimization setting (here the vector p is an 1-vector, so it is an integer between 1 and n):

- The initial graph denoted $G_\alpha(V_\alpha, E_\alpha)$ is obtained by adding a single vertex x to V, with weight α, and connecting this new vertex to every vertex in V. Thus, $V_\alpha = V \cup \{x\}$, and $E_\alpha = E \cup \bigcup_{v_i \in V}(x, v_i)$. In this graph, a trivial optimum independent set is $\{x\}$. This trivial solution will be the initial optimum used in the reoptimization instance.

- The perturbed graph $G'_\alpha(V'_\alpha, E'_\alpha)$ is obtained by adding a single vertex y to G_α, also with weight α, and connecting this new vertex to vertex x only.

Denote by OPT' an optimal independent set in G'_α. Notice that y (whose weight is α) can be added to an optimal independent set in H (whose weight is also α) to produce a feasible solution in G'_α, so that: $w(\mathrm{OPT}') \geqslant 2\alpha$.

Now, suppose that one runs the approximation algorithm `A` on the so-obtained reoptimization instance I_α. By hypothesis on `A`, it holds that $w(S_\alpha) \geqslant (1/2 + \varepsilon)w(\mathrm{OPT}') \geqslant (1 + \varepsilon)\alpha$.

Considering the lower bound on its weight, we can assert that the solution returned by `A`, does not contain x (the only independent set containing x is x itself, and thus it cannot have weight more than α). Moreover, it must contain y, otherwise it would be restricted to an independent set in G, so it couldn't have weight more than α. So, it holds that $w(S_\alpha[V]) = w(S_\alpha) - w(y) \geqslant (1 + \varepsilon)\alpha - \alpha = \varepsilon\alpha$, where $w(S_\alpha[V])$ denotes the restriction of S_α to the initial graph H.

Now, consider the following approximation algorithm `A1` for MAX INDEPENDENT SET:

> build n reoptimization instances I_i in the same way as I_α (only the weights of vertices x and y will be different from one instance to the other), for $i = 1, \ldots, n$, and run the reoptimization `A` on each of them. Denoting by S_i the solution returned by `A` on instance I_i, and $S_i[V]$ its restriction to the initial graph H, output the set $S_{\max}[V]$ with maximal weight among $S_i[V]$'s.

Obviously, considering that $1 \leqslant \alpha \leqslant n$, it holds that $S_{\max}[V] \geqslant S_\alpha[V] \geqslant \varepsilon\alpha$. Thus, the algorithm `A1`, using n times the algorithm `A` as subroutine, produces in polynomial time an ε-approximation for UNWEIGHTED MAX INDEPENDENT SET, which is impossible unless $\mathbf{P} = \mathbf{NP}$. ▮

Note that the results also hold when a constant number h of vertices are inserted. Indeed, it is easy to see that all the arguments of the proof remain valid when the set of inserted vertices is $\{y_1, \ldots, y_h\}$ each with weight α/h and connected only to vertex x.

Proposition 2. *Under insertion of one vertex and unless* $\mathbf{P} = \mathbf{NP}$*,* MAX INDEPENDENT SET *is not approximable within ratio* $(1/2 + (1/(n-1)^{\epsilon})$*, for any* $\epsilon > 0$*, where* n *is the order of the perturbed graph.*

Let us note that inapproximability bounds stated in Propositions 4, 6, 9, 11 and 14, that are of the form $\rho+\varepsilon$, $\varepsilon \in (0,1)$, can be strengthened to $\rho+n^{-\varepsilon}$. Indeed, the proofs of these propositions are based upon the argument that the existence of a $(\rho+\varepsilon)$-approximation algorithm for a given reoptimization problem $R\Pi$ induce the existence of a $O(\varepsilon)$-approximation algorithm for the "static" support Π of $R\Pi$. However, the "static" problems dealt in these propositions are not only inapproximable within $O(\varepsilon)$, unless $\mathbf{P} = \mathbf{NP}$, but within $O(n^{-\varepsilon})$. Hence, revisiting their proofs, one can replace ε by $n^{-\varepsilon}$ getting so inapproximability bounds $\rho + n^{-\varepsilon}$ instead.

3.2 MAX k-COLORABLE SUBGRAPH

Given a graph $G(V, E, w)$ and a constant $k \leqslant n$, the MAX k-COLORABLE SUBGRAPH problem consists of determining the maximum-weight subset $V' \subseteq V$ that induces a subgraph of G that is k-colorable.

The result of Section 3.1 can be generalized to the MAX k-COLORABLE SUBGRAPH problem as shows the following proposition.

Proposition 3. *In the reoptimization setting, under insertion of one vertex,* MAX k-COLORABLE SUBGRAPH *is inapproximable within ratio* $\frac{k}{k+1} + \varepsilon$ *in polynomial time, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. As before, we will start by considering a reoptimization approximation algorithm `A` for MAX k-COLORABLE SUBGRAPH, which, in polynomial time, computes a solution with approximation ratio bounded by $\frac{k}{k+1} + \varepsilon$. Once more, we consider a graph $H(V, E)$, where all vertices have weight 1. Denoting by OPT an optimal k-colorable subgraph in H, and taking into account that a k-colorable subgraph can be divided in k independent sets, it holds that $w(\mathrm{OPT}) \leqslant k\alpha$.

We now build a reoptimization instance I_α of MAX k-COLORABLE SUBGRAPH as follows:

- The initial graph G_α is obtained by adding to H a clique X of k vertices $X = (x_1, \ldots, x_k)$, each with weight α, and connecting all these vertices to all vertices of the initial graph H. In this graph, an optimal k-colorable subgraph is given by the k-clique X, and this solution will be the initial optimum used in the reoptimization instance.
- The perturbed graph G'_α is obtained by adding a single vertex y to G_α, also with weight α, and connecting this new vertex to all vertices in X.

Denote by OPT' an optimal k-colorable subgraph in G'_α. Notice that y, and $k-1$ vertices of X (with weight, in all $k\alpha$) can be added to an optimal independent set in H (with weight α) to produce a feasible solution in G'_α, so that: $w(\mathrm{OPT}') \geqslant (k+1)\alpha$.

Let S_α be the solution returned by `A` on the reoptimization instance I_α we just described. It holds that:

$$w(S_\alpha) \geqslant \left(\frac{k}{k+1} + \varepsilon\right) w(\mathrm{OPT}') \geqslant (1+\varepsilon)k\alpha \tag{1}$$

As before, noticing that this weight is strictly superior to the weight of the initial optimum, we can assert that the new vertex y belongs to the solution S_α. Now, denote by j the number of vertices of X that belong in S_α, $0 \leqslant j \leqslant k-1$. The restriction of S_α to H, that is $S_\alpha[V]$ forms a $(k-j)$-colorable subgraph, and its weight verifies:

$$w(S_\alpha[V]) = w(S_\alpha) - w(S_\alpha \cap X) - w(y) = w(S_\alpha) - (j+1)\alpha$$

Combination of the expression above with (1), leads to $w(S_\alpha[V]) \geqslant (k - j - 1 + k\varepsilon)\alpha$.

Now, recall that an optimal solution for MAX k-COLORABLE SUBGRAPH cannot have more than $k\alpha$ vertices. Then, approximation ratio between the set $w(S_\alpha[V])$ (which is a feasible solution for the MAX k-COLORABLE SUBGRAPH problem in G) and the unknown optimum OPT on the graph $H(V,E)$ is bounded as follows:

$$\frac{w(S_\alpha[V])}{w(\text{OPT})} \geqslant \frac{(k-j-1+k\varepsilon)\alpha}{k\alpha} \tag{2}$$

Finally, taking into account that $j \leqslant k-1$, (2) can be simplified, and leads to $\frac{w(S_\alpha[V])}{w(\text{OPT})} \geqslant \varepsilon$.

As before, a polynomial approximation algorithm ensuring an approximation ratio bounded by ε for MAX k-COLORABLE SUBGRAPH consists of building n reoptimization instances I_i ($1 \leqslant i \leqslant n$), running the algorithm `A` on each of them, and outputting the biggest set $S_{\max}[V]$).

However, MAX k-COLORABLE SUBGRAPH has been proved to be inapproximable within any ratio $n^{-\varepsilon}$ for some $\varepsilon \in (0,1)$ [24, 27], which forbids the existence of any constant-approximation algorithm for this problem, unless $\mathbf{P} = \mathbf{NP}$. ▮

Once more, the result can be generalized to the case where h vertices are inserted. However, the inapproximability bound gets lower for bigger values of h. To be even more precise, it holds that:

Proposition 4. *In the reoptimization setting, under the insertion of h vertices,* MAX k-COLORABLE SUBGRAPH *is inapproximable within ratio* $\max\left\{\frac{k}{k+h}, \frac{1}{2}\right\} + \varepsilon$ *in polynomial time, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. Consider a graph H, instance of MAX INDEPENDENT SET, and denote by α the independence number associated with this instance. We build in polynomial time a graph H_k, instance of MAX k-COLORABLE SUBGRAPH, by duplicating k times the instance of MAX INDEPENDENT SET, and connecting all pairs of vertices from different copies. The graph associated with this instance is denoted $H_k(V,E)$. In H_k, an optimal k-colorable subgraph has weight exactly $k\alpha$. Indeed, given the structure of instance H_k, it holds that its independence number is the same as H (namely α), so that no k-colorable subgraph can weight more than $k\alpha$. On the other hand, taking an optimal independent set in each copy produces a k-colorable subgraph with weight exactly $k\alpha$. We build a reoptimization instance $I_{\alpha,k,h}$ of MAX k-COLORABLE SUBGRAPH as follows:

- The initial graph $G_{\alpha,k,h}$ is obtained by adding to the graph H_k a clique X of k vertices $X = (x_1, \ldots, x_k)$, each with weight α, and connecting all these vertices to all vertices of V. In this graph, an optimal k-colorable subgraph is given by the k-clique X, and this solution will be the initial optimum used in the reoptimization instance.

- The perturbed graph $G'_{\alpha,k,h}$ is obtained by adding a clique Y of h vertices $y_1, \ldots, y_h$ to $G_{\alpha,k,h}$, also with weight α, and connecting all these vertices to all vertices in X.

Denote by OPT′ an optimal k-colorable subgraph in $G'_{\alpha,k,h}$. An optimal k-colorable subgraph in Y has weight at most $\min\{h,k\}\alpha$, and, considering that Y is disconnected from V, the union of a k-colorable subgraph in Y and one in V is also k-colorable. Recall that an optimal k-colorable subgraph in V has weight $k\alpha$, then it holds that $w(\text{OPT}') \geqslant (k + \min\{h,k\})\alpha$.

Denote by `A` a $\max\left\{\frac{k}{k+h}, \frac{1}{2}\right\} + \varepsilon$ approximation algorithm for the reoptimization of MAX k-COLORABLE SUBGRAPH under the insertion of h vertices. Let $S_{\alpha,k,h}$ be the solution returned by `A` on the reoptimization instance $I_{\alpha,k,h}$ we just described. It holds that:

$$w(S_{\alpha,k,h}) \geqslant \left(\frac{k}{k+\min\{h,k\}} + \varepsilon\right) \text{OPT}' \geqslant (1+\varepsilon)k\alpha$$

However, considering that $X \cup Y$ is a clique on $k+h$ vertices, each with weight α , then the restriction of $S_{\alpha,k,h}$ to $X \cup Y$ cannot have weight more than $k\alpha$. Hence, $w(S_{\alpha,k,h}[V]) = w(S_{\alpha,k,h}) - w(S_{\alpha,k,h}[X \cup Y] \geqslant \varepsilon k \alpha$.

Now, notice that $S_{\alpha,k,h}[V]$ is a partitioned in at most k independent sets, the biggest of which has weight at least $\varepsilon\alpha$, and is constrained to be included in a single copy of the original instance of MAX INDEPENDENT SET. Thus, building n reoptimization instances $I_{i,k,h}$ ($1 \leqslant i \leqslant n$), and applying algorithm A on each of them, one can find in polynomial time an independent set of size at least $\varepsilon\alpha$ in the original instance of MAX INDEPENDENT SET, which is impossible unless $\mathbf{P} = \mathbf{NP}$. ∎

This inapproximability bound is tight for the MAX INDEPENDENT SET problem (which can also be defined as the MAX 1-COLORABLE SUBGRAPH), where an easy reoptimization algorithm produces solutions with approximation ratio bounded by 1/2. We now show that this tightness holds also for MAX k-COLORABLE SUBGRAPH for any $k \geqslant 1$.

Proposition 5. *Under the insertion of h vertices,* MAX k-COLORABLE SUBGRAPH *problem is* $\left(\max\left\{\frac{k}{k+h}, \frac{1}{2}\right\}\right)$*-approximable.*

Proof. Consider a reoptimization instance I of the MAX k-COLORABLE SUBGRAPH problem. The initial graph is denoted by $G(V,E)$, and the perturbed one by $G'(V',E')$ where $V' = V \cup \{Y\}$, $Y = y_1, \ldots, y_h$. Let OPT and OPT′ denote optimal k-colorable graphs on G, and G' respectively. The initial optimum OPT is given by a set of k independent sets: $(S_1, \ldots, S_k)$, and w.l.o.g., suppose $w(S_1) \geqslant w(S_2) \geqslant \ldots, \geqslant w(S_k)$. Now, consider the following algorithm:

- if $h \geqslant k$, then apply the algorithm R1, described in [3] (ensuring a 1/2-approximate solution for any hereditary problem);
- else ($h < k$), let $\mathrm{SOL}_1 = \left(\bigcup_{i=1}^{k-h} S_i\right) \cup \{Y\}$, and $\mathrm{SOL}_2 = \mathrm{OPT}$;
- return the best solution SOL between SOL_1 and SOL_2.

First, considering that the restriction of OPT′ to V cannot define a better solution than OPT, it holds that:

$$w(\mathrm{SOL}_2) = w(\mathrm{OPT}) \geqslant w(\mathrm{OPT}') - w(Y) \tag{3}$$

Note that SOL_1 is a feasible solution. Indeed, $\bigcup_{i=1}^{k-h} S_i$ induces a $(k-h)$-colorable subgraph, thus, adding h vertices to it (here, the set Y) induces a k-colorable subgraph.

Moreover, $w\left(\bigcup_{i=1}^{k-h} S_i\right) \geqslant \frac{k-h}{k} w(\mathrm{OPT}) \geqslant \frac{k-h}{k}\left(w(\mathrm{OPT}') - w(Y)\right)$, which leads to:

$$w(\mathrm{SOL}_1) \geqslant \frac{k-h}{k}\big(w(\mathrm{OPT}') - w(Y)\big) + w(Y) \geqslant \frac{k-h}{k} w(\mathrm{OPT}') + \frac{h}{k} w(Y) \tag{4}$$

Finally, summing (3) and (4) with coefficients 1 and k/h, one gets:

$$w(\mathrm{SOL}_2) + \frac{k}{h} w(\mathrm{SOL}_1) \geqslant \frac{k}{h} w(\mathrm{OPT}')$$

and taking into account that $\frac{k+h}{h} w(\mathrm{SOL}) \geqslant w(\mathrm{SOL}_2) + \frac{k}{h} w(\mathrm{SOL}_1)$, it holds that $w(\mathrm{SOL}) \geqslant \frac{k}{k+h} w(\mathrm{OPT}')$, and the proof is completed. ∎

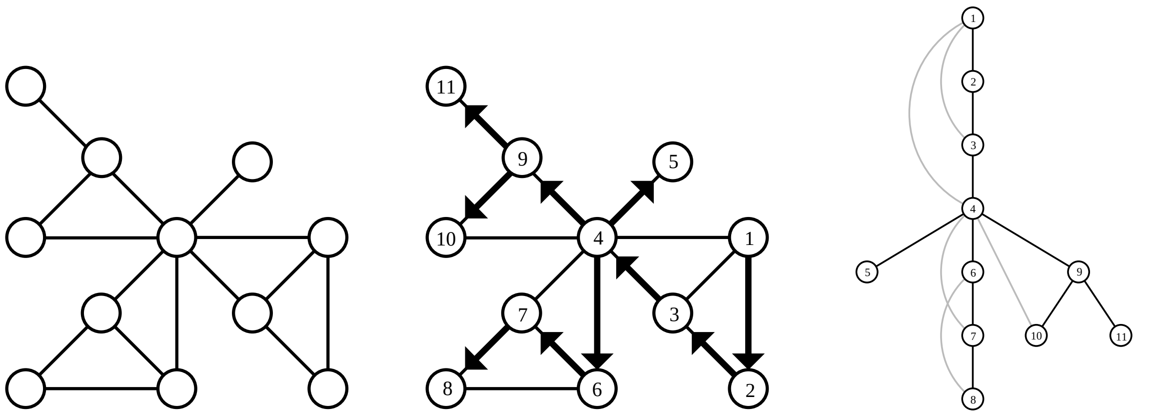
3.3 MAX P_k-FREE SUBGRAPH

A graph is said to be P_k-free if it does not contain a path on k edges. Here, P_k-free means that the graph does not admit a P_k as *minor* (and not as *induced subgraph*). For example, a C_{k+1} admits a P_k as minor, so, although it does not admit a P_k as induced subgraph, we consider that a C_{k+1} is not P_k-free. Hence, the MAX P_k-FREE SUBGRAPH problem discussed in this subsection refers to MAX PARTIAL P_k-FREE SUBGRAPH (with P_k as forbidden minor), and not to MAX INDUCED P_k-FREE SUBGRAPH (with P_k as forbidden induced subgraph). Formally, the MAX P_k-FREE SUBGRAPH problem handled in this section consists, given a graph $G(V, E, w)$ and a constant $k \leqslant n$, of finding a maximum-total weight set of vertices that induces a subgraph of G that is P_k-free. Before proceeding to the approximability analysis of MAX P_k-FREE SUBGRAPH, we need to prove the following lemma.

Lemma 1. *A P_k-free graph can be colored with k colors in polynomial time.*

Proof. Let G be a P_k-free graph, and consider the following algorithm (if G is not connected, run the procedure on each of its connected components):

- Starting from an arbitrary vertex of the graph, label all vertices from v_1 to v_n following their order of appearance in a DFS on G.
- Starting from v_1, which receices color S_1, assign to each vertex the color labelled with their depth in the tree T induced by the DFS (vertices of depth 2 will be assigned to color S_2, etc).



(a) A P_7-free graph

(b) the DFS labelling

(c) Tree T induced by the DFS

Figure 1: A k-coloring of a P_k-free graph

An example is provided in Figure 1 on a P_7-free graph. We show that following this procedure always produces a feasible k-coloring.

Consider the rooted tree T induced by the DFS on G ($T \subseteq G$). Of course, in T, some edges are missing with respect to G. However, this tree shows some interesting properties. First, notice that T has a depth bounded by k (otherwise, a path on $k+1$ vertices would exist in T and thus in G). Moreover, no pair of vertices on the same level in T can be neighbors in G. Indeed, consider two vertices labelled v_i and v_j in T (w.l.o.g. $i < j$), that are neighbors in G. It is easy to see that the subtree rooted at v_i contains all vertices that are reachable from v_i in the subgraph induced by vertices $v_{i+1}, \ldots, v_n$. Thus, if i and j are neighbors in G, then j is bound to be part of this subtree, and to be labelled with color at least S_{i+1}. Thus, the subset of vertices of a given depth i in T is always an independent set in G, and assigning a distinct color to each of these sets results in a feasible k-coloring.

Note that such a coloring is the best one can get in the general case, since the worst case is a K_k (a clique of k vertices), which is a P_k-free graph, and whose chromatic number is k. ∎

Proposition 6. *Under one vertex insertion,* MAX P_k-FREE SUBGRAPH *is inapproximable within ratio* $\frac{2k}{3k+1} + \varepsilon$ *in polynomial time if* k *is odd, and inapproximable within ratio* $\frac{2k}{3k+2} + \varepsilon$ *if* k *is even, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. Consider a graph H that has independence number α, on which one wishes to solve (or approximate) MAX INDEPENDENT SET problem. Transform this graph H into a graph $H_k(V, E)$ in the following way:

- each vertex v_i of H is turned into a clique V_i of k vertices in H_k
- if (v_i, v_j) belongs to H, then all edges between vertices of cliques V_i and V_j are in E.

Considering that the biggest independent set in H has size α, then the same holds in H_k: an independent set in G can only take one vertex in each clique V_i, and it cannot take two vertices from cliques that corresponds to neighbors in H.

Following Proposition 1, it holds that any P_k-free subgraph in H_k can be partitioned in k independent sets, so that, denoting by OPT an optimal P_k-free subgraph in H_k, $|\text{OPT}| \leqslant k\alpha$.

Moreover, denoting by IS an optimal independent set in H (that has exactly α vertices), then, in H_k, the union of all cliques corresponding to vertices of IS defines a P_k-free subgraph with value exactly $k\alpha$, so that $|\text{OPT}| = k\alpha$.

Remark 1. Following the same arguments, it holds that for any $i \leqslant k$ a P_i-free subgraph in H_k cannot weight more than $i\alpha$ (and an optimal one has weight exactly $i\alpha$). ∎

First, suppose that k is odd. We build a weighted reoptimization instance $I_{\alpha,k}$ of MAX P_k-FREE SUBGRAPH as follows:

- The initial graph $G_{\alpha,k}$ is obtained by adding to the graph H_k a set of vertices X which consists of two cliques X_1, and X_2 to G. Both these cliques have k vertices, each with weight $k\alpha$. X_1, is divided into two subcliques X_{1C} and X_{1NC}. X_{1C} has $(k+1)/2$ vertices that are all connected to all vertices in V, while the other$(k-1)/2$ vertices of X_{1NC} are not connected to any vertex in V. X_2 is divided in the same fashion. Finally, each vertex in V receives weight $k+1$, hence, for any $1 \leqslant i \leqslant k$ an optimal P_i-free subgraph inside V has weight exactly $i(k+1)\alpha$.
- The perturbed graph $G'_{\alpha,k}$ is obtained by adding a single vertex y to G_α, with weight $k(k+1)\alpha$, that is connected to all vertices of X.

Figure 2 provides a representation of the general structure.

In the initial graph, it holds that any P_k-free subgraph S has weight no more than $2k^2\alpha$. We distinguish the following two cases.

Case 1: $\mathbf{S} \cap \mathbf{V} = \emptyset$. In this case, $S \subseteq X$ so that $w(S) \leqslant w(X) = 2k^2\alpha$

Case 2: $\mathbf{S} \cap \mathbf{V} \neq \emptyset$. Denote by i the length of the longest path in $S \cap V$ (in terms of vertices). Then $S \cap V$ defines a P_i-free subgraph in G, which following Remark 1 has weight at most $i(k+1)\alpha$. Denote by i_1 and i_2 the number of vertices taken by S in X_{1C} and X_{2C}

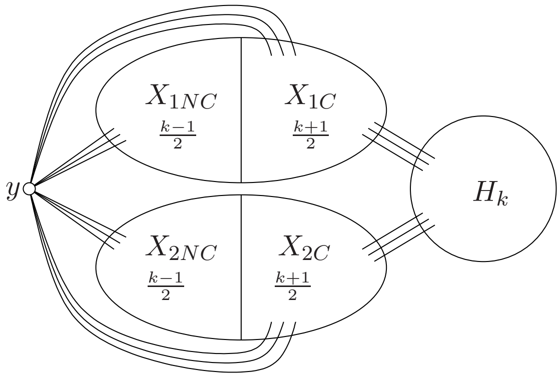


Figure 2: Reoptimization instance $I_{\alpha,k}$

respectively. Then it holds that S can take at most $k-i-i_1$ vertices in X_{1NC} and $k-i-i_2$ vertices in X_{2NC} (taking more vertices induces a P_k in S). Thus:

$$\begin{aligned} w(S) &\leqslant i(k+1)\alpha + (i_1+i_2)k\alpha + (2k-2i-i_1-i_2)k\alpha \\ &\leqslant i(k+1)\alpha + 2(k-i)k\alpha = \left(2k^2 - i(k-1)\right)\alpha \ \leqslant\ 2k^2\alpha \end{aligned}$$

that concludes Case 2.

Hence, any solution that has weight $2k^2\alpha$ is optimal and we will thus consider X (which is feasible and has weight exactly $2k^2\alpha$) as the initial optimum of the reoptimization instance I_α.

Now, notice that, denoting by F^* an optimal P_k-free subgraph in H_k, $\{y\}\cup X_{1NC}\cup X_{2NC}\cup F^*$ is a feasible solution in $G'_{\alpha,k}$. Indeed, $\{y\}\cup X_{1NC}\cup X_{2NC}$ has exactly k vertices (thus, it defines a P_k-free subgraph), and it is disconnected from H_k, so that adding an optimal P_k-free subgraph in H_k to it produces a P_k-free subgraph that has two connected components. Thus, $w\left(\{y\}\cup X_{1NC}\cup X_{2NC}\cup F^*\right) = k(3k+1)\alpha$, and $w(\mathrm{OPT}') \geqslant k(3k+1)\alpha$.

Consider a $(\frac{2k}{3k+1}+\varepsilon)$-approximation algorithm `A` for the reoptimization version of MAX P_k-FREE SUBGRAPH, and denote by $S_{\alpha,k}$ a solution returned by `A` on the reoptimization instance $I_{\alpha,k}$ we just described. Then:

$$w(S_{\alpha,k}) \geqslant \left(\frac{2k}{3k+1}+\varepsilon\right) w(\mathrm{OPT}') \geqslant (1+\varepsilon)2k^2\alpha \tag{5}$$

As before, this solution has to take y since its weight is strictly better than the initial optimum. However, if it takes y, then it cannot take more than $k-1$ vertices in $X_1\cup X_2$ so that:

$$w\left(S_{\alpha,k}[X\cup\{y\}]\right) \leqslant k(k+1)\alpha + (k-1)k\alpha = 2k^2\alpha \tag{6}$$

It immediately results from (5) and (6) that $w(S_{\alpha,k}[V]) \geqslant 2k^2\varepsilon\alpha$, where $S_{\alpha,k}[V]$ denotes the restriction of $S_{\alpha,k}$ to V. This set defines a P_k-free subgraph in G, that can be partitioned in k independent sets, the biggest of which has weight at least $2k\varepsilon\alpha$, and thus at least $2\frac{k}{k+1}\varepsilon\alpha \geqslant \varepsilon\alpha$ vertices (recall that each vertex in V has weight $k+1$).

Considering that H_k has n cliques, each corresponding to a vertex of the original graph H, any independent set in H_k must have its vertices in different cliques that do not correspond to neighbors in H. Thus, an independent set of size $\varepsilon\alpha$ in $G_{\alpha,h}$ defines an independent set of the same size in H.

Hence, when confronted to a graph H where one wants to approximate MAX INDEPENDENT SET, one could build all instances $I_{i,k}$ $(1\leqslant i\leqslant n)$, run the algorithm `A` on each of them, and find an independent set of size at least $\varepsilon\alpha$ in H_k (and thus in H) when running it on the instance $I_{\alpha,k}$. The whole procedure is polynomial, and provides a constant approximation ratio for MAX INDEPENDENT SET, which is impossible unless $\mathbf{P} = \mathbf{NP}$. Thus, the first part of the Proposition 6 is true.

Now, suppose that k is even. Let $\text{opt} = k\alpha$, fix an integer t with arbitrarily large value, and consider the following reoptimization instance $I_{\text{opt},k}$:

- The initial graph $G_{\text{opt},k}$ is obtained by adding t cliques $X_1, \dots, X_t$ to H_k. The set X is defined as $X = \bigcup_{i=1}^{t} X_i$. All these cliques have k vertices, each with weight $\frac{\text{opt}}{kt-1}$. All the vertices in X are connected to all vertices in V. All vertices in V have weight 1.

- The perturbed graph $G'_{\text{opt},k}$ is obtained by adding a single vertex y to $G_{\alpha,k}$, with weight $\frac{(k/2+1)t-1}{kt-1}\text{opt}$, that is connected to all vertices of the t cliques $X_1, \dots, X_t$.

We prove that a P_k free subgraph S in the initial graph $G_{\text{opt},k}$ has weight at most $\frac{kt}{kt-1}\text{opt}$. For this, we still distinguish two cases.

Case 1: $\mathbf{S} \cap \mathbf{V} = \emptyset$. In this case, $w(S) \leqslant w(X) = \frac{kt}{kt-1}\text{opt}$.

Case 2: $\mathbf{S} \cap \mathbf{V} \neq \emptyset$. Denote by i the length of the longest path in $S[V]$ in terms of vertices ($1 \leqslant i \leqslant k$). Put differently, $S[V]$ is P_i-free but not P_{i-1}-free. Thus, following Remark 1:

$$w(S[V]) \leqslant i \times \alpha = \frac{i}{k}\text{opt} = \frac{i \times t}{k \times t}\text{opt} < \frac{i \times t}{k \times t - 1}\text{opt} \tag{7}$$

On the other hand, no set $S[X_j]$ can contain more than $k - i$ vertices, otherwise, there would exist a path on $k + 1$ vertices in $S[X_j \cup V]$. Moreover, denoting the size of the biggest clique in $S[X]$ by $c_{\max}$, ($0 \leqslant c_{\max} \leqslant k - 1$), and the size of the second biggest clique in $S[X]$ by $c_{\max 2}$, one verifies that $c_{\max 2}$ cannot exceed $k - i - c_{\max}$. Indeed, consider the path going through the biggest clique in X then through the i vertices of longest path in $S[V]$, and finally through the second biggest clique in X. By hypothesis, this path cannot go through more than k vertices, so that: $c_{\max} + i + c_{\max 2} \leqslant k$. Thus, regarding the values of $c_{\max}$ and $c_{\max 2}$ it holds that:

$$c_{\max} \leqslant k - i - c_{\max_2} \tag{8}$$

and taking into account that $c_{\max} \geqslant c_{\max 2}$, it also holds that:

$$c_{\max_2} \leqslant \left\lfloor \frac{k-i}{2} \right\rfloor \tag{9}$$

Indeed, suppose that $c_{\max_2} \geqslant \lfloor \frac{k-i}{2} \rfloor + 1$, then $c_{\max} + i + c_{\max 2} \geqslant k + 1$, which is impossible.

Notice that $S[X]$ consists of t cliques, one with $c_{\max}$ vertices, and $t - 1$ with at most $c_{\max_2}$ vertices, the total number of vertices in $S[X]$ is bounded as follows:

$$\begin{aligned}
|S[X]| &\leqslant c_{\max} + (t-1)c_{\max 2} \\
&\leqslant k - i - c_{\max_2} + (t-1)c_{\max 2} \\
&\leqslant k - i + (t-2)\left\lfloor \frac{k-i}{2} \right\rfloor
\end{aligned} \tag{10}$$

where the second line follows from (8), and the third from (9).

Recall that each vertex in X has weight exactly $\frac{\text{opt}}{kt-1}$, then one can finally bound the weight of $S[X]$:

$$\begin{aligned} w(S[X]) &\leqslant \frac{\text{opt}}{kt-1}\left(k-i+(t-2)\lfloor\frac{k-i}{2}\rfloor\right) \\ &\leqslant \frac{t\text{opt}}{kt-1}\left(\lfloor\frac{k-i}{2}\rfloor+\frac{k-i-2\lfloor\frac{k-i}{2}\rfloor}{t}\right) \\ &\leqslant \frac{t\text{opt}}{kt+1}+\left(k-i+\frac{k-i-2\lfloor\frac{k-i}{2}\rfloor-t\lceil\frac{k-i}{2}\rceil}{t}\right) \quad (11) \\ &\leqslant \frac{(k-i)t}{kt+1}\text{opt} \quad (12) \end{aligned}$$

where (12) follows from noticing that for a big enough value of t, the rightmost fraction in (11) is negative.

Finally, taking into account that V and X define a partition of the graph $G_{\text{opt},k}$, and combining (7) and (12), one proves that for any value of i, $w(S) = w(S[V]) + w(S[X]) \leqslant \frac{kt}{kt-1}\text{opt}$ that concludes Case 2.

Hence, any feasible solution that has weight exactly $\frac{kt}{kt-1}\text{opt}$ on the initial graph $G_{\text{opt},k}$ is optimal, and we can consider X as the initial optimum.

Now, notice that $y \cup F^*$ (where F^* denotes an optimal P_k-free subgraph in H_k) is a feasible solution in the modified graph $G'_{\text{opt},k}$. Thus, denoting by OPT' an optimal solution in $G'_{\text{opt},k}$, it holds that $w(\text{OPT}') \geqslant w(y \cup F^*) = \frac{\left(\frac{k}{2}+1\right)t-1}{kt-1}\text{opt} + \text{opt} = \frac{\frac{3k+2}{2}t-2}{kt-1}\text{opt}$.

Now, suppose that there exists an reoptimization algorithm `A` that ensures an approximation ratio bounded by $\frac{2k}{3k+2} + \varepsilon$ for the reoptimization version of MAX P_k-FREE SUBGRAPH. Then, denoting by $S_{\text{opt},k}$ a solution returned by this algorithm on the reoptimization instance $I_{\text{opt},k}$:

$$\begin{aligned} w(S_{\text{opt},k}) &\geqslant \left(\frac{2k}{3k+2}+\varepsilon\right)\frac{\frac{3k+2}{2}t-2}{kt-1}\text{opt} \\ &\geqslant \frac{kt}{kt-1}\text{opt}\left(1+\varepsilon+\frac{\varepsilon(t(k+2)/2-2)-1}{kt}\right) \quad (13) \\ &\geqslant (1+\varepsilon)\frac{kt}{kt-1}\text{opt} \quad (14) \end{aligned}$$

where (14) follows from noticing that the rightmost fraction in (13) is positive for a big enough value of t.

Considering that this solution is strictly better than the initial optimum (which has weight exactly $\frac{kt}{kt-1}\text{opt}$), then it has to take y. For arguments similar to those explained in Case 2 of the present analysis (with $i = 1$), $S_{\text{opt},k}[X]$ contains at most $k-1+(t-2)\lfloor\frac{k-1}{2}\rfloor$ vertices (the result is derived from (10)). Recall that k is supposed to be even, and that each vertex in X has weight $\frac{\text{opt}}{kt+1}$, then $w(S_{\text{opt},k}[X]) \leqslant k-1+(t-2)\left(\frac{k}{2}-1\right)\frac{\text{opt}}{kt-1}$ $\leqslant \left(\left(\frac{k}{2}-1\right)t+1\right)\frac{\text{opt}}{kt-1}$, from which one immediately derives $w(S_{\text{opt},k}[X \cup \{y\}]) = w(S_{\text{opt},k}[X]) + w(y) \leqslant \frac{kt}{kt-1}\text{opt}$.

Thus, the weight of the restriction of $S_{\text{opt},k}$ to vertices of V, denoted by $S_{\text{opt},k}[V]$, is bounded by $w(S_{\text{opt},k}[V]) \geqslant \varepsilon\frac{kt}{kt-1}\text{opt} \geqslant \varepsilon\text{opt} = \varepsilon k\alpha$.

Hence, for reasons similar to those explained in the proofs of all inapproximability bounds, the existence of a reoptimization polynomial algorithm for MAX P_k-FREE SUBGRAPH providing ratio $\frac{2k}{3k+1} + \varepsilon$ (for odd values of k) or $\frac{2k}{3k+2} + \varepsilon$ (for even values of k) induces the existence of an

ε polynomial approximation algorithm for MAX INDEPENDENT SET (in the original graph H), which is impossible unless $\mathbf{P} = \mathbf{NP}$. ∎

We now prove that this ratio is tight for small values of k, namely when $k \leqslant 6$. Consider the following hypothesis:

Hypothesis 1. In polynomial time, a P_k-free subgraph S can be partitioned in 3 sets S_1, S_2 and S_3, such that both S_1 and S_2 are $P_{\lceil k/2 \rceil - 1}$-free, and $w(S_3) \leqslant w(S)/k$ if k is odd and $w(S_3) \leqslant 2w(S)/k$ if k is even. ∎

For values of k where Hypothesis 1 is true, the following proposition holds.

Proposition 7. *Under Hypothesis 1,* MAX P_k-FREE SUBGRAPH *problem is approximable within ratio $\frac{2k}{3k+1}$ for odd values of k, and $\frac{2k}{3k+2}$ for even values of k in the reoptimization setting, under one vertex insertion.*

Proof. Consider a reoptimization instance I of MAX P_k-FREE SUBGRAPH, given by two graphs G (initial) and G' (perturbed) and an optimal solution OPT on the initial graph G. G and G' differ only by vertex y (and its incident edges) which belongs to G' but not to G. Classically, denoting by OPT$'$ an optimal solution on G', it holds that $w(\mathrm{OPT}) \geqslant w(\mathrm{OPT}') - w(y)$. Suppose that Hypothesis 1 is verified, and consider the following algorithm:

- partition OPT in 3 sets S_1, S_2 and S_3 as defined in Hypothesis 1, and without loss of generality, suppose $w(S_1) \geqslant w(S_2)$; set $\mathrm{SOL}_1 = S_1 \cup \{y\}$ and $\mathrm{SOL}_2 = \mathrm{OPT}$;
- return the best solution SOL between SOL_1 and SOL_2.

First, let us prove that this algorithm returns a feasible solution: SOL_2 is trivially feasible in G', and consider a path P in SOL_1. If this path does not go through y then it cannot go through more than $\lceil k/2 \rceil - 1$ vertices (by hypothesis, S_1 is $P_{\lceil k/2 \rceil - 1}$-free). If it does go through y, then denote by P_1 the set of vertices visited before y in P and P_2 the set of vertices visited after. Considering that both P_1 and P_2 are included in S_1, which is supposed to be $P_{\lceil k/2 \rceil - 1}$-free, then $|P_1|, |P_2| \leqslant \lceil k/2 \rceil - 1$, so that $|P| = |P_1| + |P_2| + 1 \leqslant k$, and thus SOL_1 is also P_k-free.

We define r_k as an integer that is equal to 1 if k is odd, and to 2 if k is even. Regarding the partitioning induced by Hypothesis 1, it holds that $w(S_3) \leqslant \frac{r_k w(\mathrm{OPT})}{k}$, and thus $w(S_1) \geqslant (w(\mathrm{OPT}) - w(S_3))/2 \geqslant \frac{k-r_k}{2k} w(\mathrm{OPT})$, thus:

$$w(\mathrm{SOL}_2) \geqslant \frac{k-r_k}{2k} w(\mathrm{OPT}) + w(y) = \frac{k-r_k}{2k} w(\mathrm{OPT}') + \frac{k+r_k}{2k} w(y) \tag{15}$$

On the other hand:

$$w(\mathrm{SOL}_1) \geqslant w(\mathrm{OPT}') - w(y) \tag{16}$$

Summing (15) and (16) with coefficients 1 and $\frac{k+r_k}{2k}$ respectively, one finally proves that:

$$\frac{3k+r_k}{2k} w(\mathrm{SOL}) \geqslant w(\mathrm{SOL}_2) + \frac{k+r_k}{2k} w(\mathrm{SOL}_1) \geqslant w(\mathrm{OPT}')$$

that concludes the proof. ∎

Now let us prove that Hypothesis 1 is true for $k \leqslant 6$ (note that it is an open question whether it remains true for bigger values of k, or not). In what follows, we suppose that the P_k-free graph S to be partitioned is connected (if it is not so, then proving that the hypothesis is true for each of its connected components amounts to proving it for the whole graph).

$\mathbf{k = 1, 2.}$ Simply set $S_3 = S$ and $S_1, S_2 = \emptyset$.

k = 3. Split the graph in three independent sets S'_1, S'_2,and S'_3 in polynomial time (that is possible according to Lemma 1), and w.l.o.g., suppose $w(S'_1) \geqslant w(S'_2) \geqslant w(S'_3)$. Finally, set $S_1 = S'_1$, $S_2 = S'_2$, and $S_3 = S'_3$.

k = 4. Split the graph in four independent sets from S'_1 to S'_4 in polynomial time, and w.l.o.g., suppose $w(S'_1) \geqslant w(S'_2) \geqslant w(S'_3) \geqslant w(S'_4)$. Finally, set $S_1 = S'_1$, $S_2 = S'_2$, and $S_3 = S'_3 \cup S'_4$.

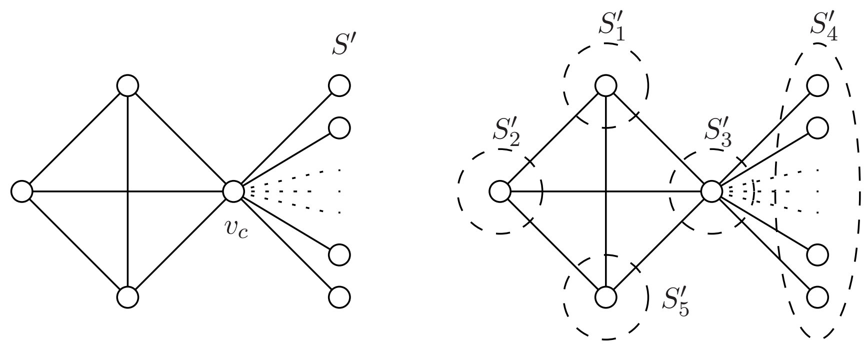
k = 5. Here, we wish to prove that S can be partitioned in three sets S_1, S_2 and S_3 such that both S_1 and S_2 are P_2-free and $w(S_3) \leqslant w(S)/5$. We distinguish the following five cases.

Case 1. S contains a C_5 (a cycle on 5 vertices). In this case, S contains exactly 5 vertices. Let $v_{\min}$ denote the vertex with minimum weight among the 5 vertices of S, and set $S_3 = \{v_{\min}\}$. Finally assign two remaining vertices to each of S_1 and S_2 arbitrarily.

Case 2. S contains no C_5 but it contains a C_4 with two diagonals (a clique on 4 vertices). In this case let S' denote the set of vertices of S that do not belong to this K_4. Then it holds that:

- S' forms an independent set. Indeed, suppose that there exist an edge v_i, v_j in S', then there must exist a path through at least 6 vertices in S, that would go through v_i, v_j, and the K_4 (recall that S is connected).

- All vertices of S' are connected to the same vertex in the K_4, say v_c. Indeed, in a K_4, any pair of vertices can be the two extremities of a P_4. Suppose that vertices of S' are connected to two different vertices in the K_4, say v_{c1} and v_{c2}. First, notice that a single vertex cannot be connected to both v_{c1} and v_{c2}, since it would form a C_5 which contradicts the hypothesis of Case 2. Then, if at least two different vertices of S' are connected to two different vertices v_{c1} and v_{c2}, then there exists a path through 6 vertices in G, impossible.



(a) A P_5, C_5-free graph containing a K_4

(b) A partition in 5 final independent sets

Figure 3: Case 2

Thus, the structure of Case 2 can be only be as described in Figure 3(a). In this case, one can partition the graph in 5 independent sets from S'_1 to S'_5 (as shown in Figure 3(b)), such that $S \setminus S'_{\min}$ (where $S'_{\min}$ denotes the lightest of the 5 independent sets) can be partitioned in two P_2-graphs: it suffices to ensure that S'_3 and S'_4 are not in the same partition.

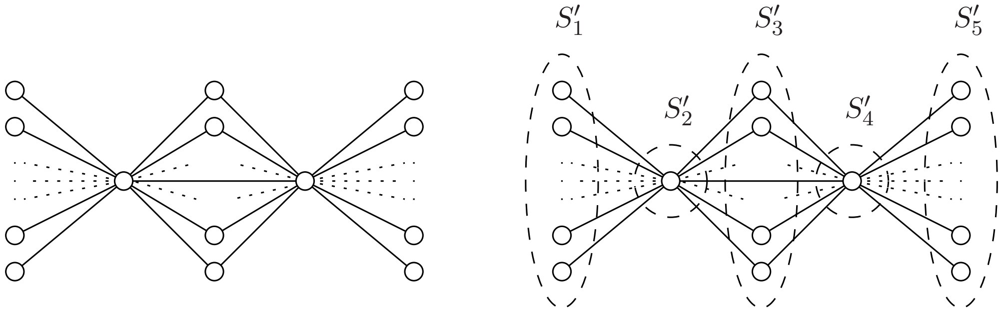
Case 3. S contains no C_5 but it contains a C_4 with at most one diagonal. Again, let S' denote the set of vertices of S that do not belong to this C_4. For reasons similar to those explained in Case 2 it holds that:

- S' is an independent set.

- Two different vertices of S' cannot be connected to two vertices that are neighbors in the C_4 (because inside the C_4, there always exists a path going through 4 vertices, and having two neighbors in the C_4 as extremal vertices).

- Two different vertices of S' cannot be connected to two vertices of the gadget that are opposite in the C_4, but not connected by the diagonal, if any (for similar reasons).

Hence, vertices of S' can be either all connected to the same vertex in the C_4 (and can be partitioned in the same fashion as in Case 2), or connected to two opposite vertices in it (and only to the two opposites that are connected by the diagonal, if any). The structure is represented in Figure 4(a), and once more, it can be partitioned in five independent sets from S'_1 to S'_5 (Figure 4(b)). Here, a partitionning in three sets S_1, S_2 and S_3 that fit Hypothesis 1 is very easy to obtain: set $S_1 = S'_1 \cup S'_3 \cup S'_5$, $S_2 = S'_2 \cup S'_4$, and $S_3 = \emptyset$.



(a) The overall structure ...

(b) ... and its partition in 5 final independent sets

Figure 4: A K_3 and a C_4 with three vertices in common

Case 4. S contains no C_4, and no C_5, but it contains a C_3. Let v_1, v_2 and v_3 denote the three vertices of the C_3, and let $S' = S \setminus \{v_1, v_2, v_3\}$. Then it holds that there is no path from v_i to v_j ($1 \leqslant i < j \leqslant 3$) using at least one vertex of S', otherwise there would exist a C_4 in S, which contradicts the hyposthesis of Case 4. Thus, S' can be partitioned in 3 disconnected subgraphs H_1, H_2 and H_3, that are respectively reachable by v_1 in $S' \cup v_1$, by v_2 in $S' \cup v_2$ and by v_3 in $S' \cup v_3$ (Figure 5).

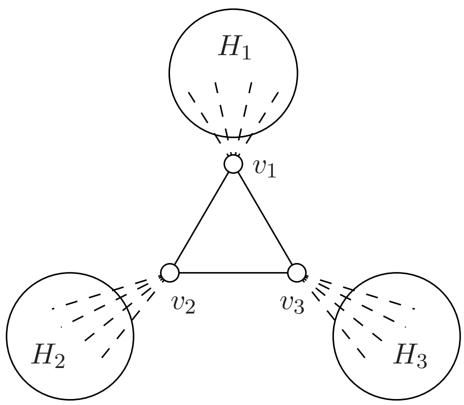


Figure 5: General structure of S in Case 4

Case 4-a. Suppose that there exists at least one edge (h_1, h_2) in one set H_i, say in H_1. Then it holds that either h_1 or h_2 is connected to v_1, otherwise there would exist a path going through 6 vertices in S: $(v_3, v_2, v_1, P, h_1, h_2)$ or $(v_3, v_2, v_1, P, h_2, h_1)$ (where P contains at least one vertex). So, w.l.o.g, suppose that v_1 is connected to h_1. Then there exists at least 2 path through exactly 5 vertices in S: $(v_3, v_2, v_1, h_1, h_2)$, and $(v_2, v_3, v_1, h_1, h_2)$, so that any vertex in H_2 or H_3 immediately results in a path through 6 vertices, impossible. Hence, $H_2 = \emptyset$ and $H_3 = \emptyset$.

Considering the symmetry of the structure, one can assert that if there exist edges in S', then they are all included in one H_i while the other two are empty. Continue to suppose that these edges are within H_1. These edge (h_i, h_j) can be of two types:

- Type 1: both h_i and h_j are connected to v_1. These edges necessarily form a matching. Indeed, suppose that two of these edges have one vertex in common, say (h_i, h_j) and (h_i, h_k) then there exists a path through 6 vertices in S: $(v_2, v_3, v_1, h_j, h_i, h_k)$.

- Type 2: only h_i is connected to v_1. For similar reasons, these edges form disconnected stars rooted at neighbors of v_1.

Note also that edges of Type 1 and 2 cannot have a vertex in common.

In this case, a partitionning of the vertices of S in 5 independent sets from S'_1 to S'_5 can be obtained in the following way:

- the first independent set S'_1 contains only v_1,

- $S'_2 \cup S'_3$ forms a perfect matching between all pairs of vertices forming a triangle along with v_1 (including v_2 and v_3). Put differently, the cut between and S'_2 and S'_3 contains all edges of Type 2 as well as (v_2, v_3),

- S'_4 contains all remaining neighbors of v_1,

- S'_5 contains all neighbors of vertices in S'_4.

The partition is presented in Figure 6, and one can verify that for any $S'_{\min} \in (S_1, \ldots, S_5)$, $S \setminus S'_{\min}$ can be partitioned in two P_2-free subgraphs, each using two remaining independent sets of the partition. For example, if the lightest independent set is S'_4, the partition is (S'_2, S'_3) (that is a perfect matching), and (S'_1, S'_5) (that is an independent set), or if the lightest independent set is S'_5, the partition is (S'_2, S'_3, S'_4) (that is a perfect matching and an independent set), and (S'_1) (that is a single vertex).

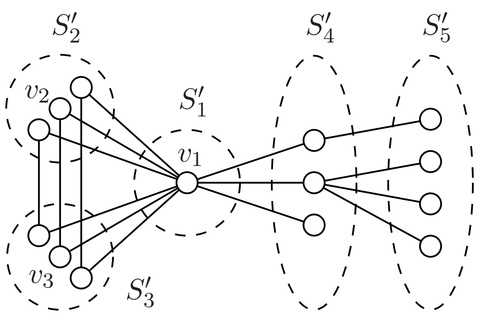


Figure 6: Partition in Case 4-b

Case 4-b. S' is an independent set. This case is much easier in the sense that the graph S can immediately be partitioned into two P_2-free subgraphs: $S_1 = H_1 \cup \{v_2\} \cup \{v_3\}$ and $S_2 = H_2 \cup H_3 \cup v_1$ (and $S_3 = \emptyset$).

Case 5. S contains no C_5, no C_4, and no C_3. Hence, S is a tree, so that it is trivially 2-colorable, thus defining S_1 as the first color, S_2 as the second color, and S_3 as an empty set, one gets a partitioning of S that fits Hypothesis 1.

In any case, Hypothesis 1 is verified when $k = 5$.

k = 6. In this case, Hypothesis 1 is verified if S can be partitioned in 3 P_2-free subgraphs (the lightest of which will be S_3, and the other two S_1 and S_2). Here, we distinguish the following four cases.

Case 1. S contains a C_6. It is clear that S contains exactly 6 vertices, which can easily be partitioned in three P_2-free subgraphs (2 vertices in each set of the partition).

Case 2. S contains a C_5 and no C_6. Let S' denote the set of vertices that are not in C_5. It holds that S' is an independent set, and that a pair of vertices that are neighbors in the C_5 cannot both have neighbors in S' (if one of these two properties is not verified, then there exists a P_6 in S. Thus, without making assumption on the number of chords in the C_5, the general structure of S is as described in Figure 7, with a partitioning in three P_2-free subgraphs (represented in the figure as white, grey, and black vertices).

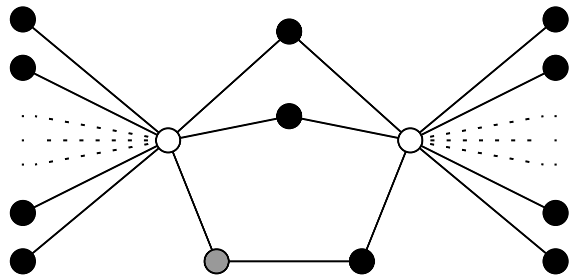


Figure 7: Partition of P_6-free graph in 3 P_2-free subgraphs in Case 2

Case 3. S contains a C_4. We distinguish here the following two subcases.

Case 3-a. S contains a C_4 but no C_5 and no C_6, and denoting by S' the set of vertices that are not in the C_4, S' contains at least an edge (v_1, v_2). Obviously, these two vertices v_1 and v_2 cannot be connected to two different vertices of the C_4, otherwise, S would contain a C_5 or a C_6.

Moreover, at least one of these two vertices must be connected to a vertex of the C_4. Indeed, taking into account that S is supposed to be connected, if none of these vertices is connected to a vertex of the C_4, then there exists a P_6 in S.

Finally, supposing w.l.o.g that v_1 is the vertex connected to a vertex x in the C_4, then no neighbor of x in the C_4 can be connected to any vertex of S' (otherwise there would exist a P_6 in S).

Hence, denoting vertices of the C_4 by x, y, z and t as in Figure 8, only vertices x and z might have neighbors in S'. Moreover, denoting by $N(\{x\} \cup \{z\})$ the neighborhood of these two vertices, it holds that this set is P_2 free, otherwise a path on 7 vertices would exist in S (the 3 vertices of the P_2 in $N(\{x\} \cup \{z\})$, and the 4 vertices of the C_4). Thus, coloring in white the vertices of $N(\{x\} \cup \{z\})$ as well as y and t, and in black all the other vertices of S, one gets a partition of S into two P_2 free colors.

Case 3-b. S contains a C_4 but no C_5 and no C_6, and let S' denote the set of vertices that are not in the C_4, S' contains no edge. This case is much simpler than the previous one, considering that S' forms an independent set. To get a partition of S into three P_2-free subgraphs, it suffices

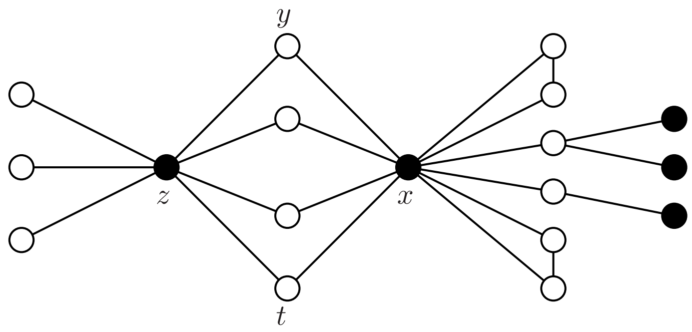


Figure 8: Partition of P_6-free graph in 2 P_2-free subgraphs in Case 3-a

to consider S' itself as the first subset of the partition, $\{x\} \cup \{z\}$ as the second, and $\{y\} \cup \{t\}$ as the third.

Case 4. S contains no C_4, no C_5 and no C_6. In this case, it holds that the neighborhood $N(x)$ of any vertex x is P_2 free, since a P_2 (on 2 edges, and 3 vertices) in $N(x)$ amounts to a C_4 in $x \cup N(x)$. Moreover, denoting by $N(N(x))) = N^2(x))$ the set of neighbors of vertices in $N(x)$, it holds that $N(N(x))$ is also P_2-free, since the neighborhood of each vertex in $N(x)$ is P_2-free (considering what we stated above), and disjoint from one another (otherwise there would exist C_4 in S). Naturally, the same holds for any $N^i(x)$.

Hence, in this case, the graph can be easily partitioned in 2 P_2-free subgraphs: starting from a arbitrary vertex that is colored black, one colors its neighborhood white, then the neighborhood of its neighborhood black, etc. Considering what we proved earlier, each color defines a P_2-free subgraph.

Finally, Hypothesis 1 is also verified when $k = 6$.

3.4 MAX SPLIT SUBGRAPH

Given a graph $G(V, E, w)$, the MAX SPLIT SUBGRAPH problem consists of determining a maximum-weight subset $V' \subseteq V$ that induces a split subgraph of G. A split graph is a graph whose vertices can be partitioned into two sets C and S, C being a clique, and S being an independent set. Any subset of a clique remains a clique, and any subset of independent set remains an independent set, hence, being a split graph is a hereditary property. Moreover, considering that the property is false for a complete bipartite graph with at least two vertices in each independent set, the result of Theorem 1 applies to the MAX SPLIT SUBGRAPH problem. So MAX SPLIT SUBGRAPH is inapproximable within any constant ratio, unless $\mathbf{P} = \mathbf{NP}$.

We prove that this strong inapproximability result does not hold in the reoptimization setting, but we first need to prove the following lemma.

Lemma 2. *Let G be a graph with $h \leqslant 3$ vertices. It holds that $w(G_S) + w(G_C) \geqslant \frac{h+1}{h} w(G)$ if $h \leqslant 2$ and $w(G_S) + w(G_C) \geqslant \frac{5}{4} w(G)$ if $h = 3$, where G_S and G_C respectively denote an optimal independent set and an optimal clique in G.*

Proof. If G is a clique, then $w(G_C) = w(G)$ and $w(G_S) \geqslant \frac{w(G)}{h}$, and symmetrically, if G is an independent set, then $w(G_S) = w(G)$ and $w(G_C) \geqslant \frac{w(G)}{h}$. In both cases, the proposition is verified.

If G is neither a clique nor an independent set (which might occur when $h = 3$), then there are only two possible configurations, both represented in Figure 9.

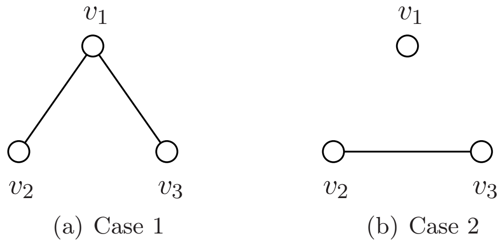


Figure 9: Graphs G that are neither cliques nor independent sets

In Case 1, there are two maximal cliques: $\{v_1, v_2\}$ and $\{v_1, v_3\}$, so $w(G_C) \geqslant \frac{2w(v_1)+w(v_2)+w(v_3)}{2}$. On the other hand there are two maximal independent sets: $\{v_1\}$ and $\{v_2, v_3\}$, so $w(G_S) \geqslant \frac{1}{4}w(v_1) + \frac{3}{4}(w(v_2) + w(v_3))$. In all: $w(G_C) + w(G_S) \geqslant \frac{5w(v_1)+5w(v_2)+5w(v_3)}{4} = \frac{5}{4}w(G)$.

Taking into account the symmetry between Cases 1 and 2, the same bound holds for Case 2, which concludes the proof. ∎

Proposition 8. *Under insertion of h vertices,* MAX SPLIT SUBGRAPH *problem is $\frac{h+1}{2h+1}$-approximable for $h \leqslant 2$, and $\frac{5}{9}$-approximable for $h = 3$.*

Proof. Consider a reoptimization instance I of the MAX SPLIT SUBGRAPH problem. The initial graph is denoted by $G(V, E)$, and the perturbed one $G'(V', E')$, where $V' = V \cup Y$ where $|Y| = h \leqslant 3$. Let OPT and OPT$'$ denote optimal split-graphs on G, and G' respectively. The initial optimum OPT is given by a clique C and an independent set S. Let Y_S and Y_C denote optimal independent sets and cliques in Y. Consider the following algorithm:

- Let $\text{SOL}_1 = S \cup Y_C$, $\text{SOL}_2 = C \cup Y_S$, and $sol_3 = \text{OPT}$
- return the best solution SOL among SOL_1, SOL_2, and SOL_3.

First, noticing that $S \cup Y_C$ and $C \cup Y_S$ both define split graphs, it holds that the algorithm returns a feasible solution. Then summing $w(\text{SOL}_1)$, and $w(\text{SOL}_2)$, we get the following equality:

$$w(\text{SOL}_1) + w(\text{SOL}_2) = w(C) + w(S) + w(Y_C) + w(Y_S) \geqslant \begin{cases} w(\text{OPT}) + \frac{h+1}{h}w(Y) & \text{if } h \leqslant 2, \\ w(\text{OPT}) + \frac{5}{4}w(Y) & \text{if } h = 3. \end{cases}$$

The second line follows noticing that $w(C) + w(S) = w(\text{OPT})$, and taking into account that, according to Lemma 2, $w(Y_S) + w(Y_C) \geqslant \frac{h+1}{h}w(Y)$ if $h \leqslant 2$, and $w(Y_S) + w(Y_C) \geqslant \frac{5}{4}w(Y)$ if $h = 3$.

Notice that $w(\text{OPT}) \geqslant w(\text{OPT}') - w(Y)$, one verifies that:

$$w(\text{SOL}_1) + w(\text{SOL}_2) \geqslant \begin{cases} w(\text{OPT}') + \frac{1}{h}w(Y) & \text{if } h \leqslant 2 \\ w(\text{OPT}') + \frac{1}{4}w(Y) & \text{if } h = 3 \end{cases} \tag{17}$$

$$w(\text{SOL}_3) \geqslant w(\text{OPT}') - w(Y) \tag{18}$$

Finally, summing (17) and (18) with coefficients h and 1, if $h \leqslant 2$, and 4 and 1 if $h = 3$:

$$\begin{cases} (2h+1)w(\text{SOL}) \geqslant h(w(\text{SOL}_1) + w(\text{SOL}_2)) + w(\text{SOL}_3) \geqslant (h+1)w(\text{OPT}') & \text{if } h \leqslant 2 \\ 9w(\text{SOL}) \geqslant 4(w(\text{SOL}_1) + w(\text{SOL}_2)) + w(\text{SOL}_3) \geqslant 5w(\text{OPT}') & \text{if } h = 3 \end{cases}$$

and the proof is completed. ∎

Recall that for any h (and a fortiori for $h \geqslant 4$) the problem is 1/2-approximable by the algorithm `R1` presented in [3]. We prove that these simple approximation algorithms achieve the best constant ratios possible.

Proposition 9. *Under vertex insertion,* MAX SPLIT SUBGRAPH *is inapproximable within ratios* $\frac{h+1}{2h+1} + \varepsilon$ *when* $h \leqslant 2$, $\frac{5}{9} + \varepsilon$ *when* $h = 3$, *and* $\frac{1}{2} + \varepsilon$ *when* $h \geqslant 4$ *in polynomial time, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. Consider an unweighted graph H where one wishes to solve MAX SPLIT SUBGRAPH. Denote by α its independence number, β its clique number. Now, suppose that one builds a graph $H_{\alpha,\beta}$ by adding to H two graphs H_1 and H_2, that are complementary graphs of H, and connecting all vertices in H to vertices in H_2, while H_1 is disconnected from H. Between H_1 and H_2, vertices are connected in the following way:

- if $\alpha < \beta$, then all vertices in H_1 are connected to all vertices in H_2,
- if $\alpha \geqslant \beta$, H_1 and H_2 are completely disconnected.

In $H_{\alpha,\beta}$, both the clique and independence numbers are $\alpha + \beta$. Indeed, Notice that each clique in H becomes an independent set in both H_1 and H_2, and vice versa.

If a clique C takes at least one vertex in H_1, then, either $\alpha \geqslant \beta$ and this clique must be included in H_1 ($|C| \leqslant \alpha$), or $\alpha < \beta$, and this clique can have vertices in H_1 and H_2 but not in H, so that $|C| \leqslant 2\alpha < \alpha + \beta$. On the other hand, if C takes no vertex in H_1, then it must be composed of a clique in H (of size at most β) and a clique in H_2 (of size at most α), so that the clique number of $H_{\alpha,\beta}$ is exactly $\alpha + \beta$.

By symmetry, the same holds for the independence number.

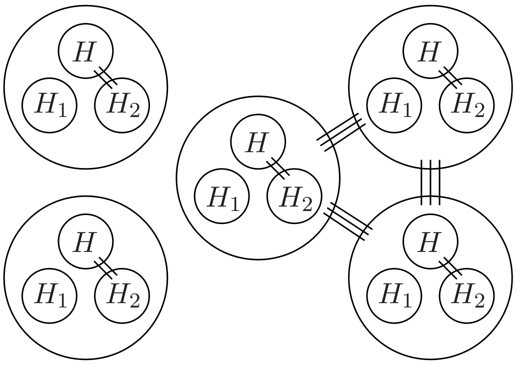


Figure 10: The graph $H_{\alpha,\beta,3,3}$ ($\alpha \geqslant \beta$)

In what follows, we note $\gamma = \alpha + \beta$. Consider the graph H_{α,β,h_1,h_2} ($h_1, h_2 \geqslant 1$) that consists of $h_1 + h_2 - 1$ copies of the graph $H_{\alpha,\beta}$. Among them, h_1 copies are disjoint from one another, and h_2 copies are connected by all possible pairs of vertices from different copies (an example is provided in Figure 10). It holds that:

- H_{α,β,h_1,h_2} has independence number $h_1\gamma$,
- H_{α,β,h_1,h_2} has clique number $h_2\gamma$,
- An ε-approximate solution for MAX SPLIT SUBGRAPH in H_{α,β,h_1,h_2} can be easily derived into a $(2\varepsilon/3)$-approximate solution for MAX SPLIT SUBGRAPH in H. Indeed, any solution SG on H_{α,β,h_1,h_2} can be partitioned into $3(h_1+h_2-1)$ feasible solutions on H, the biggest of

which has weight at least $(2\varepsilon/3)\gamma$ when SG is an ε-approximate solution. Thus, MAX SPLIT SUBGRAPH is inapproximable within any constant ratio in H_{α,β,h_1,h_2}, unless $\mathbf{P} = \mathbf{NP}$.

In what follows, we will build three different reoptimization instances, for cases $h \leqslant 2$, $h = 3$ and $h \geqslant 4$. In the three cases, the initial graphs will have the same generic structure, defined as G_{α,β,h_1,h_2} (h_1 and h_2 will take different values in the three specific cases).

This graph is obtained by adding to H_{α,β,h_1,h_2} a set X of vertices, which consists of a clique X_C of size $h_2\gamma + 1$ and an independent set X_S of size $h_1\gamma + 1$. Each vertex in the graph receives weight 1. X_C is disconnected from V, and each vertex of X_S is connected to all other vertices of the graph, namely vertices of both V and X_C. In this initial graph, it holds that a split-graph SG has weight at most $(h_1 + h_2)\gamma + 2$. We distinguish the three following cases.

Case 1. $SG \cap V = \emptyset$, then $w(S) \leqslant w(X) \leqslant (h_1 + h_2)\gamma + 2$.

Case 2. $|SG \cap V| = 1$, and denote by v the single vertex of $SG \cap V$. In this case, SG cannot take more than $\max\{h_1, h_2\}\gamma + 1$ vertices in X. Indeed, if it takes more vertices, then it means that SG contains at least two vertices of each set X_S and X_C, which necessarily forms a forbidden subgraph along with v. It means that $w(SG) \leqslant \max\{h_1, h_2\}\gamma + 2 \leqslant (h_1 + h_2)\gamma + 2$.

Case 3. $|SG \cap V| \geqslant 2$. Denote by SG_C the clique in SG and by SG_S the independent set in SG.

If $SG[V]$ is a clique, then the clique in SG, SG_C, can contain at most $h_2\gamma$ vertices in V and one vertex in X_S, so that $w(SG_C) \leqslant h_2\gamma + 1$. On the other hand, the biggest independent set in X is X_S, so that $w(SG_S) \leqslant w(S_X) \leqslant h_1\gamma + 1$. In all:

$$w(SG) = w(SG_S) + w(SG_C) \leqslant (h_1 + h_2)\gamma + 2$$

A symmetrical arguments holds when $SG[V]$ is an independent set.

Suppose now that $SG[V]$ is neither a clique nor an independent set. Then $w(SG[V]) \leqslant (h_1 + h_2)\gamma$, and both SG_S and SG_C contain at least one vertex in V. Thus, SG_C cannot contain any vertex of X_C, and at most one of vertex of X_S. Symmetrically, SG_S cannot contain any vertex of X_S, and at most one of vertex of X_C. In all, $w(SG[X]) \leqslant 2$, and once more, $w(SG) \leqslant (h_1 + h_2)\gamma + 2$, concluding so Case 3.

Thus, any solution that has weight exactly $(h_1 + h_2)\gamma + 2$ is optimal in G_{α,β,h_1,h_2}, so in all reoptimization instances we will build, we can consider X as the initial optimum.

Assume $h \leqslant 2$. We build a reoptimization instance, $I_{\alpha,\beta,h}$ in the following way:

- The initial graph is the graph $G_{\alpha,\beta,h,1}$. We proved earlier that X is an optimum on this graph. Here, its weight is $(h+1)\gamma + 2$.
- The perturbed graph $G'_{\alpha,\beta,h,1}$ is obtained by adding a set of vertices Y to $G_{\alpha,\beta,h,1}$, which consists of an independent set of h vertices, each with weight γ. All vertices in Y are connected to all vertices in X_S only.

The overall structure is represented in Figure 11, as well as the weight of optimal independent sets and cliques (denoted by S^* and C^* in the Figure) in all sets V, X_C, X_S and Y.

Notice that, in the perturbed graph $Y \cup X_C \cup IS^*$ (where IS^* is an optimal independent set in V) defines an split graph of weight $(2h+1)\gamma + 1$. Indeed, $Y \cup IS^*$ defines an independent set, while X_C defines an clique. Thus, denoting by OPT' an optimal split graph in $G'_{\alpha,\beta,h,1}$, it holds that $w(\text{OPT}') \geqslant (2h+1)\gamma + 1$.

Suppose that, for a given $h \leqslant 2$, there exists an approximation algorithm `A` for the reoptimization version of MAX SPLIT SUBGRAPH, which provides an approximation ratio bounded by $\frac{h+1}{2h+1} + \varepsilon$, under the insertion of h vertices. Denoting by $S_{\alpha,\beta,h}$ a solution returned by this

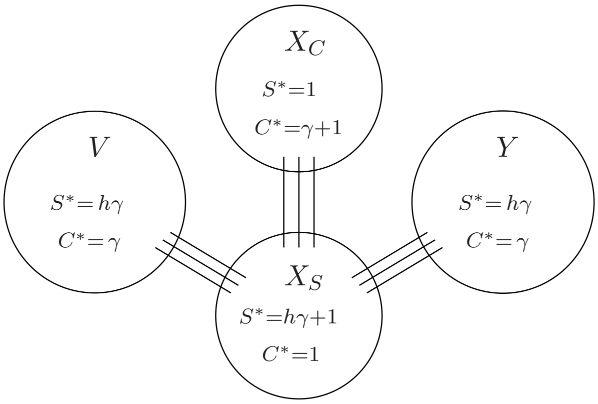


Figure 11: The reoptimization instance $I_{\alpha,\beta,h}$, $h \leqslant 2$

algorithm on the reoptimization instance $I_{\alpha,\beta,h}$ we just described, it holds that:

$$w(S_{\alpha,\beta,h}) \geqslant \left(\frac{h+1}{2h+1} + \varepsilon\right) \mathrm{OPT}' \geqslant (1+\varepsilon)(h+1)\gamma$$

However, a split graph SG in $X \cup Y$ (and a fortiori the restriction of $S_{\alpha,\beta,h}$ to $X \cup Y$, denoted by $S_{\alpha,\beta,h}[X \cup Y]$) cannot have weight more than $(h+1)\gamma + 2$. We distinguish here the following two cases.

Case 1. SG takes at most one vertex in X_S, then $w(SG[X_S]) \leqslant 1$, and thus:

$$w(SG) = w(SG[X_S]) + w(X_C) + w(Y) \leqslant 1 + \gamma + 1 + h\gamma = (h+1)\gamma + 2$$

Case 2. SG takes at least two vertices in X_S, then the independent set in SG can contain only vertices of X_S. In other words, the vertices of $Y \cup X_C$ can only be part of the clique in SG. It is quite obvious that the biggest clique in $Y \cup X_C$ is X_C itself so that in this case $w(SG) \leqslant w(X) = (h+1)\gamma + 2$. One immediately derives from this result that $w(S_{\alpha,\beta,h}[X \cup Y]) \leqslant (h+1)\gamma + 2$ and Case 2 is concluded.

So, in both cases it is verified that $w(S_{\alpha,\beta,h}[V]) = w(S_{\alpha,\beta,h}) - w(S_{\alpha,\beta,h}[X \cup Y]) \geqslant \varepsilon\gamma - 2$. Considering that γ is not a constant, if an algorithm `A` exists, one can get in polynomial time a constant-approximate solution for MAX SPLIT SUBGRAPH in the graph $H_{\alpha,\beta,h,1}$, which is impossible unless $\mathbf{P} = \mathbf{NP}$.

Suppose now that $h = 3$. In this case, the reoptimization instance $I_{\alpha,\beta,h}$ is built as follows:

- The initial graph is the graph $G_{\alpha,\beta,2,3}$. X is an optimum on this graph. Here, its weight is $5\gamma + 2$.
- The perturbed graph $G'_{\alpha,\beta,2,3}$ is obtained by adding a path Y to $G_{\alpha,\beta,2,3}$, which has 3 vertices, y_1, y_2 and y_3. y_1 is the central vertex of the path, and has weight 2γ, while y_2 and y_3 are end vertices of the path, and have both weight γ. All vertices in Y are connected to all vertices in X_S, but not to any vertex in X_C. y_1 is connected to all vertices in V, while vertices y_2 and y_3 are not connected to any vertex in V.

Notice that, in the perturbed graph $Y \cup F^*$ (where F^* is an induced optimal split subgraph in V) defines a split graph of weight at least $9\gamma - 1$ (in V the maximum clique F^*_C and the maximum independent set F^*_S might have one vertex in common). Indeed, $y_1 \cup F^*_C$ defines a clique, while $y_2 \cup y_3 \cup F^*_S$ defines an independent set. Thus, denoting by OPT' an optimal split graph in $G'_{\alpha,\beta,h}$, it holds that $w(\mathrm{OPT}') \geqslant 9\gamma - 1$.

Assume that there exists an approximation algorithm A for the reoptimization version of MAX SPLIT SUBGRAPH, which provides an approximation ratio bounded by $\frac{5}{9}+\varepsilon$, under the insertion of $h = 3$ vertices. Denoting by $S_{\alpha,\beta,h}$ a solution returned by this algorithm on the reoptimization instance $I_{\alpha,\beta,h}$ we just described, it holds that:

$$w(S_{\alpha,\beta,h}) \geqslant \left(\frac{5}{9}+\varepsilon\right)\mathrm{OPT}' \geqslant (1+9\varepsilon)5\gamma - \frac{5}{9} - \varepsilon \geqslant (1+\varepsilon)5\gamma \tag{19}$$

where the last inequality follows from noticing that $5/9+\varepsilon < 8\varepsilon\gamma$, otherwise a single vertex defines a 8ε-approximate solution in V, which is supposed to be impossible to provide in polynomial time, unless $\mathbf{P} = \mathbf{NP}$.

However, it holds that a split graph SG in $X \cup Y$ has weight at most $5\gamma + 3$ and we study the following four cases.

Case 1. $SG[Y] = \emptyset$. Then $w(SG) \leqslant w(X) = 5\gamma + 2$,

Case 2. $|SG[Y] = 1|$. If the single vertex is part of the clique in SG, then this clique can take at most one additional vertex in X_S. Symmetrically, if this vertex is part of the independent set in SG, then this independent set can take at most one additional vertex in X_C:

$$w(SG) \leqslant w(y_1) + 1 + \max\{w(X_C), w(X_S)\} \leqslant 5\gamma + 3$$

Case 3. $|SG[Y] = 2|$. If $SG[Y]$ has two connected vertices, then:

- $w(SG[Y]) = 3\gamma$;
- the clique in SG can take at most one vertex in X_S, and none in X_C;
- the independent set in SG cannot weight more than $w(X_S) = 2\gamma + 1$.

If, on the other hand, $SG[Y]$ has two disconnected vertices, then:

- $w(SG[Y]) = 2\gamma$;
- the independent set in SG can take at most one vertex in X_C, and none in X_S;
- the clique in SG cannot weight more than $w(X_C) = 3\gamma + 1$.

Hence, in both cases, one verifies that $w(SG) \leqslant 5\gamma + 3$.

Case 4. $|SG[Y] = 3|$. In this case, $w(SG[Y]) = 4\gamma$, and SG can take at most two vertices of X, one of X_S that can be part of the clique in SG, of X_C that can be part of the independent set in SG. In all, $w(SG) \leqslant 4\gamma + 2$, that concludes Case 4.

Taking into account that any split graph in $X \cup Y$ cannot have weight more than $5\gamma + 3$, the same holds a fortiori for $S_{\alpha,\beta,h}[X \cup Y]$. Combining this bound with (19), one immediately derives that:

$$w(S_{\alpha,\beta,h}[V]) \geqslant (1+\varepsilon)5\gamma - 5\gamma - 3 = \varepsilon\gamma - 3 \tag{20}$$

For reasons already explained, (20) makes impossible the existence of a polynomial algorithm A that ensures a $\frac{5}{9}+\varepsilon$ approximation ratio under the insertion of $h = 3$ vertices, unless $\mathbf{P} = \mathbf{NP}$.

Finally, suppose that $h \geqslant 4$. We build the following reoptimization instance $I_{\alpha,\beta,h}$:

- The initial graph is the graph $G_{\alpha,\beta,1,1}$. In it, X defines an optimal split graph of weight $2\gamma + 2$.
- The perturbed graph $G'_{\alpha,\beta,h}(V_{\alpha,\beta,h}, E_{\alpha,\beta,h})$ is obtained by adding a set Y of h vertices, in which 4 vertices have weight $\gamma/2$, and are as represented in Figure 12: all these vertices are connected to all vertices in X_S, while only y_1 and y_2 are connected to all vertices in V. The other $h - 4$ vertices in Y have null weight, and will be ignored in what follows.

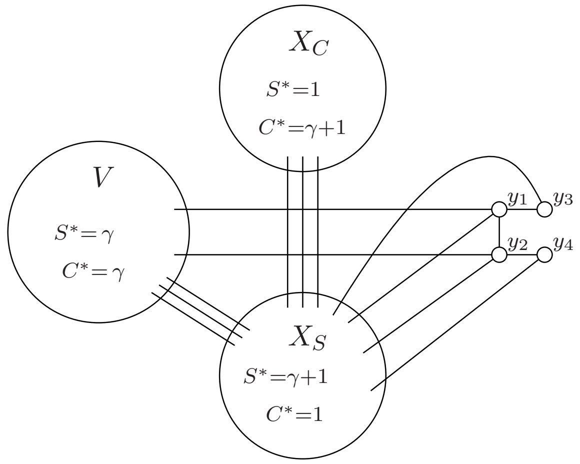


Figure 12: The reoptimization instance $I_{\alpha,\beta,h}$, $h \geqslant 4$

Notice that in V, an optimal split graph F^* has weight at least $2\gamma - 1$ (since a clique and independent set can have at most one vertex in common), notice also that $Y \cup F^*$ defines a feasible solution: the clique in F^* forms a clique along with y_1 and y_2, and the independent set in F^* forms an independent set along with y_3 and y_4. Thus, denoting by OPT' an optimal solution in the perturbed graph $G'_{\alpha,\beta,h}$, it holds that $w(\mathrm{OPT}') \geqslant 4\gamma - 1$.

Suppose that, for a given $h \geqslant 4$, there exists an approximation algorithm `A` for the reoptimization version of MAX SPLIT SUBGRAPH, which provides an approximation ratio bounded by $\frac{1}{2} + \varepsilon$, under the insertion of h vertices. Denoting by $S_{\alpha,\beta,h}$ a solution returned by this algorithm on the reoptimization instance $I_{\alpha,\beta,h}$ we just described, its weight can be bounded as follows:

$$w(S_{\alpha,\beta,h}) \geqslant \left(\frac{1}{2} + \varepsilon\right) w(\mathrm{OPT}') \geqslant (1 + 2\varepsilon)2\gamma - \frac{1}{2} - \varepsilon \geqslant (1 + \varepsilon)2\gamma \tag{21}$$

where the last inequality follows from noticing that $1/2 + \varepsilon < 2\varepsilon\gamma$. If not, then a single vertex defines a ε-approximate solution in V.

We prove now that a split graph SG in $X \cup Y$ (and a fortiori the restriction of $S_{\alpha,\beta,h}$ to this set) cannot have weight bigger than $2\gamma + 2$. For this, we study the following three cases.

Case 1. $|SG \cap Y| = 0$. Obviously $w(SG) \leqslant w(X) = 2\gamma + 2$,

Case 2. $|SG \cap Y| = 1, 2$. SG can take at most $\gamma + 2$ vertices in X, otherwise, it takes at least 2 vertices in both X_C and X_S, which form a forbidden subset along with one vertex in Y. Thus, $w(SG) \leqslant \gamma + 4 \leqslant 2\gamma + 2$

Case 3. $|SG \cap Y| \geqslant 3, 4$. Given the structure of Y, then 3 (and a fortiori 4) vertices cannot form an independent set, nor a clique. Thus, SG can take at most one vertex in each set X_C and X_S, and $w(SG) \leqslant 2\gamma + 2$ and Case 3 is concluded.

Hence, the restriction of $S_{\alpha,\beta,h}$ to $X \cup Y$ has weight bounded as follows:

$$w(S_{\alpha,\beta,h}[X \cup Y]) \leqslant \frac{2}{\gamma} + 2 \tag{22}$$

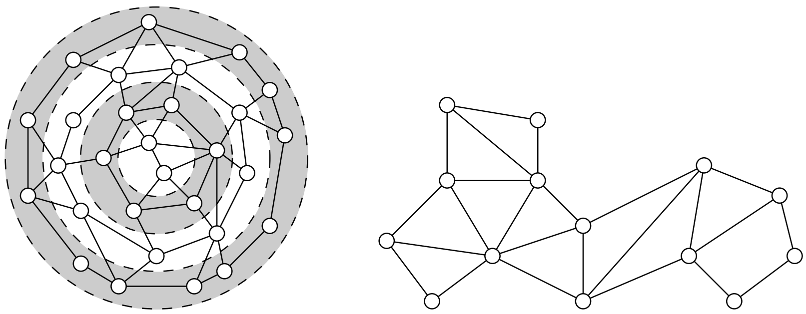
Combining (21) and (22), one easily derives:

$$w(S_{\alpha,\beta,h}[V]) \geqslant (1 + \varepsilon)2\gamma - 2\gamma - 2 = \varepsilon\gamma - 2 \tag{23}$$

So, $S_{\alpha,\beta,h}[V]$ is a constant-approximate solution for MAX SPLIT SUBGRAPH in the graph $H_{\alpha,\beta,1,1}$, which is impossible to provide in polynomial time unless $\mathbf{P} = \mathbf{NP}$. ▮

3.5 MAX PLANAR SUBGRAPH

Given a graph $G(V, E, w)$, the MAX PLANAR SUBGRAPH problem consists of determining a maximum-weight subset $V' \subseteq V$ that induces a planar subgraph of G. A planar graph is defined as a graph that can be embedded in the plane in a way that no edges cross each other. It is proved in [29] that the class of planar graph is characterized by two forbidden minors: K_5 and $K_{3,3}$. On the other hand, an outerplanar graph is a planar graph that can be embedded in the plane such that all its vertices belong to the outer-face of the embedding (see Figure 13(b)). Similarly, the class of outerplanar graphs is characterized by two forbidden minors: K_4 and $K_{2,3}$. Given a planar graph $G(V, E)$, the level of a vertex is defined inductively as follows: the set $L1$ of vertices at level 1 is constituted by vertices on the exterior face of G; then, the set L_i of vertices at level i is the set of vertices on the exterior face of the subgraph of G induced by $V\ (L_1, \ldots, L_{i-1})$. A planar graph is said to be k-outerplanar if every vertex is at level at most k (for at least one planar embedding, see Figure 13(a)). Every planar graph is k-outerplanar for some k and can be embedded in polynomial time. An interesting property regarding outerplanarity is that the addition of a single vertex to an outerplanar graph results in a planar graph. We prove the following lemma.



(a) A 4-outerplanar graph　　(b) An outerplanar graph

Figure 13: Outerplanarity and k-outerplanarity

Lemma 3. *Let $G(V, E)$ be a planar graph. The vertex set V of G can be partitioned in two subsets S_1 and S_2 such that both induced subgraphs $G[S_1]$ and $G[S_2]$ are outerplanar. Such a partition can be computed in polynomial time.*

Proof. Recall that a planar embedding of a planar graph can be obtained in polynomial time by the algorithm presented in [15]. Given an embedding of G, there is a simple way to partition G into two induced outerplanar subgraphs: recall that any planar embedding can be partitioned in layers $L_1 \ldots, L_l$, where L_1 denotes the outerface of the embedding, L_2 denotes the outerface of the embedding without L_1, L_3 the outerface of the embedding without L_1 and L_2, etc.

Notice that each layer induces an outerplanar graph, and that each layer L_i is a separating set that cuts the graph into two connected components, one that contains all layers $L_1, L_2, \ldots, L_{i-1}$, and the other that contains all layers $L_{i+1}, \ldots, L_l$.

Hence, the union of all layers with odd labels consists of disjoint layers, and is thus outerplanar, and so is the union of all layers with even labels (see also Figure 13(a)). ∎

Proposition 10. *Under vertex insertion, the* MAX PLANAR SUBGRAPH *problem is approximable within ratio* 2/3.

Proof. Consider a reoptimization instance I of the MAX PLANAR SUBGRAPH problem. The initial graph is denoted by $G(V, E)$, and the perturbed one $G'(V', E')$ where $V' = V \cup \{y\}$. Let

OPT and OPT$'$ denote optimal planar graphs on G, and G' respectively. Using the method proposed in Lemma 3, OPT can be partitioned into 2 sets S_1, and S_2 such that both these sets induce outerplanar graphs. Suppose that S_1 is the heavier of theses two sets. Consider the following algorithm:

- Let $\text{SOL}_1 = S_1 \cup \{y\}$, and $\text{SOL}_2 = \text{OPT}$;
- return the best solution SOL among SOL_1, and SOL_2.

First, notice that SOL_1 is planar, since it consists of the union of an outerplanar graph and a single vertex, so that the algorithm always returns a feasible solution for MAX PLANAR SUBGRAPH.

Then, classically, it holds that $w(\text{OPT}) \geqslant w(\text{OPT}') - w(y)$, so that $w(\text{SOL}_1)$ and $w(\text{SOL}_2)$ are bounded as follows:

$$w(\text{SOL}_1) \geqslant \frac{w(\text{OPT}') - w(y)}{2} + w(y) = \frac{w(\text{OPT}')}{2} + \frac{w(y)}{2} \tag{24}$$

$$w(\text{SOL}_2) \geqslant w(\text{OPT}') - w(y) \tag{25}$$

Summing (24) and (25) with coefficients 2 and 1, respectively, we get:

$$3w(\text{SOL}) \geqslant 2w(\text{SOL}_1) + w(\text{SOL}_2) \geqslant 2w(\text{OPT}')$$

that concludes the proof. ∎

The four colors Theorem proved in [28], claims that every planar graph is 4-colorable. We use this theorem to prove the following proposition.

Proposition 11. *Under vertex insertion, the* MAX PLANAR SUBGRAPH *problem is inapproximable within ratio* $4/5 + \varepsilon$ *in polynomial time, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. Consider a graph H that has independence number α, where one wishes to solve MAX INDEPENDENT SET problem. It is possible to turn this instance into an instance of MAX PLANAR SUBGRAPH by turning the graph H into a graph H' as follows:

- each vertex v_i of H is turned into a clique V_i of 4 vertices in H';
- if (v_i, v_j) belongs to H, then all edges between vertices of cliques V_i and V_j are in E.

For reasons similar to those explained in the proof of Proposition 6, it holds that:

- H' is also α-independent;
- denoting by OPT an optimal planar subgraph in G, then $w(\text{OPT}) = 4\alpha$.

We build a weighted reoptimization instance I_α of MAX PLANAR SUBGRAPH as follows:

- The initial graph G_α is obtained by adding a clique X of 4 vertices to H', each with weight α. E_α contains all edges between X and V.
- The perturbed graph G'_α is obtained by adding a single vertex y to G_α, with weight α that is connected to all vertices of X.

Any planar subgraph S in G_α has weight at most 4α. Indeed, let i denote the number of vertices of X in S, then $S \cap [V]$ is $(4 - i)$-colorable, so that it has weight at most $(4 - i)\alpha$. Thus, $w(S) \leqslant i\alpha + (4 - i)\alpha = 4\alpha$.

Hence, any planar subgraph that has weight exactly 4α is optimal in G_α, so we assume that X (which has weight 4α) is the initial optimum of the reoptimization instance I_α.

In G'_α, $\{y\} \cup \mathrm{OPT}$ is a feasible solution, so that $w(\mathrm{OPT}') \geqslant 5\alpha$. Consider an algorithm `A` that provides a $4/5 + \varepsilon$ for the reoptimization version MAX PLANAR SUBGRAPH. Denote by SOL_α a solution returned by `A` on the reoptimization instance I_α we just described. Then:

$$w(S_\alpha) \geqslant \left(\frac{4}{5} + \varepsilon\right) \mathrm{OPT}' \geqslant (1+\varepsilon)4\alpha$$

However, a planar subgraph in $\{y\} \cup X$ can take at most four vertices (taking 5 vertices induces a K_5), so that $w\left(S_\alpha \cap (\{y\} \cup X)\right) \leqslant 4\alpha$. Thus, it holds that $w(S_\alpha[V]) \geqslant 4\varepsilon\alpha$.

Notice that, $S_\alpha[V]$ being planar, it can be partitioned in four independent sets in polynomial time. The biggest of these 4 four independent sets has weight at least $\varepsilon\alpha$, and by construction, one can easily determine an independent set of the same size in the original MAX INDEPENDENT SET instance, which is impossible unless $\mathbf{P} = \mathbf{NP}$. ∎

4 Vertex deletion

Let us consider now the opposite kind of perturbations: vertex-deletion. When dealing with hereditary optimization problems, some properties discussed just above still remain valid, while some others do not. As before, let us consider a given instance of a hereditary problem, for which we know an optimal solution OPT. Consider now that one vertex of the graph is deleted, along with its incident edges. Two cases might occur:

- the deleted vertex y was not part of the initial optimum, so it remains the same in the new graph.
- y was part of the initial optimum, and might even have been one of its most important elements. Though having a priori no information on the quality of the initial optimum $\mathrm{OPT} \setminus \{y\}$ in the new graph G' (or rather what is left of it), we can still assert that $\mathrm{OPT} \setminus \{y\}$ remains a feasible solution in the new graph.

In what follows, we discuss to what extent the techniques used in the case of insertion can be applied to the case of deletion. As in Section 3, we will start by an inapproximability result on the most elementary hereditary problem, MAX INDEPENDENT SET (Section 4.1), and then extend the result to all inapproximable hereditary problems (Section 4.2). Then, we provide some tight positive results for MAX k-COLORABLE SUBGRAPH (Section 4.3), and finally we present general techniques for reoptimizing hereditary problems in graphs of bounded degree (Section 4.4).

4.1 MAX INDEPENDENT SET

Recall that under vertex insertion, any hereditary problem is easily 1/2-approximable. The results does not extend to the case of vertex deletion, where MAX INDEPENDENT SET is as hard to approximate as its deterministic version:

Proposition 12. *Under vertex deletion,* MAX INDEPENDENT SET *is inapproximable within any ratio $n^{-\varepsilon}$ in polynomial time, for any $\varepsilon \in (0,1)$, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. Consider an instance $H(V,E)$ of (unweighted) MAX INDEPENDENT SET, and build the following reoptimization instance I:

- The initial instance G is obtained by adding to H a single vertex y with weight n. y is connected to all the vertices in H. Obviously, the single vertex y is an optimal solution in G.

- The perturbed instance is the graph H.

Suppose that there exists an approximation algorithm A that ensures an approximation $n^{-\varepsilon}$ for MAX INDEPENDENT SET under vertex deletion, then this algorithm can be used to obtain an $n^{-\varepsilon}$ on any instance H by running A on the reoptimization instance I we just described. Hence, such an algorithm cannot exist unless **P** = **NP** [31]. ∎

It is thus clear that, unlike in the insertion case, there is no polynomial algorithm providing a constant approximation ratio for any hereditary problem in the deletion case. However, the MAX INDEPENDENT SET problem is probably the hardest to approximate in the vertex deletion setting, in the sense an optimal solution might contain a single vertex, so that the deletion of this vertex discards the whole information provided by the initial optimum.

But regarding other hereditary properties, this phenomenon will not occur systematically, and in fact, the strong inapproximability bound on the reoptimization version of MAX INDEPENDENT SET under vertex deletion cannot be transferred directly to all hereditary problems.

4.2 General negative result

When dealing with MAX INDEPENDENT SET, the whole initial optimum can disappear when deleting a single vertex, since *the minimal size of a maximal solution is 1*, put differently, a single vertex can be a maximal solution. However, this fact does not hold for any hereditary property. Consider for example the MAX BIPARTITE SUBGRAPH problem. Regarding this problem, a single vertex cannot define a maximal solution, and it takes at least two deleted vertices to delete the whole initial optimum. We derive from this idea the following general inapproximability result:

Proposition 13. *Let $M(\Pi)$ denote the minimal size of a maximal solution for a given hereditary problem Π. Under the deletion of $h \geqslant M(\Pi)$ vertices, Π is inapproximable within any ratio $n^{-\varepsilon}$ in polynomial time, unless* **P** = **NP**.

Proof. Consider an instance of a given unweighted non trivial hereditary problem Π, that consists of a graph $H(V,E)$. We build the following reoptimization instance I:

- The initial graph G is obtained by adding to H a set of vertices Y of size $h \geqslant M(\Pi)$. This set contains a gadget of size $M(\Pi)$ that constitutes a maximal solution in G, where each vertex has weight n, and $h - M(\Pi)$ vertices with weight 0 (which will be ignored in what follows).
- The perturbed graph is the graph H

It is clear that the $M(\Pi)$ vertices of weight n in Y define an optimal solution in the initial graph G: This gagdet is feasible and maximal, so that in G an optimal solution has weight at least $M(\Pi)n$. On the other hand, any solution that does not take the whole gadget has weight at most $(M(\Pi)-1)n + \text{OPT} \leqslant M(\Pi)n$, where OPT denotes the cardinality of an optimal solution in H. Thus, Y can be considered as the initial optimum of the reoptimization instance I.

Consider a reoptimization algorithm A, which, for a given $h \geqslant M(\Pi)$, does provide an approximation ratio $n^{-\varepsilon}$ under the deletion of h vertices. When using it on the reoptimization instance I we just described, this algorithm produces a $n^{-\varepsilon}$-approximate solution in H in polynomial time, which is impossible unless **P** = **NP**. ∎

Regarding the four specific problems discussed in Section 3, the four following corollaries hold.

Corollary 1. *Under deletion of $h \geqslant k$ vertices,* MAX k-COLORABLE SUBGRAPH *is inapproximable within ratio $n^{-\varepsilon}$ unless* **P** = **NP**.

Corollary 2. *Under deletion of $h \geqslant k$ vertices,* MAX P_k-FREE SUBGRAPH *is inapproximable within ratio $n^{-\varepsilon}$ unless* $\mathbf{P} = \mathbf{NP}$.

Corollary 3. *Under deletion of $h \geqslant 3$ vertices,* MAX SPLIT SUBGRAPH *is inapproximable within ratio $n^{-\varepsilon}$ unless* $\mathbf{P} = \mathbf{NP}$.

Corollary 4. *Under deletion of $h \geqslant 4$ vertices,* MAX PLANAR SUBGRAPH *is inapproximable within ratio $n^{-\varepsilon}$ unless* $\mathbf{P} = \mathbf{NP}$.

For Corollaries 1 and 2, it suffices to notices that K_k's can define maximal solution for both these problems.

For Corollary 3, notice that 3 vertices can define a maximal solution for MAX SPLIT SUBGRAPH: revisiting the proof of Proposition 13, build the gadget in Y as follows: two vertices y_1, y_2 that are connected only one to the other, and a vertex y_3 connected to all vertices in H. Clearly, $\{y_1, y_2, y_3\}$ defines a maximal (and optimal) solution in G.

Finally Corollary 4 derives from the fact that a planar graph is 4-colorable, so that a K_4 can define a maximal solution.

4.3 MAX k-COLORABLE SUBGRAPH

Following Corollary 1, it holds that no constant approximation ratio can be expected in polynomial when more than k vertices are deleted, However if the number of deleted vertices is smaller than k, the non deleted part of the initial optimum is non-empty. Following this idea we provide the following positive result for MAX k-COLORABLE SUBGRAPH:

Proposition 14. *Under deletion of $h < k$ vertices,* MAX k-COLORABLE SUBGRAPH *is approximable within ratio $\frac{k-h}{k}$.*

Proof. Consider a reoptimization instance I of the MAX k-COLORABLE SUBGRAPH problem, under deletion of $h < k$ vertices. The initial graph is denoted by $G(V, E)$, and the perturbed one $G'(V', E')$ where $G' = G[V \setminus Y]$. Let OPT and OPT$'$ denote optimal k-colorable subgraphs on G, and G' respectively. Considering that Y has h vertices, and denoting by h' the number of independent sets in OPT that contain at least one vertex of Y, it holds that $h' \leqslant h$.

Let S_1 denote the $(k - h')$-colorable subgraph in OPT which does not contain any vertex of Y, and S_2 the h'-colorable subgraph in OPT such that each independent set in it has at least one vertex in Y. Consider the simple reoptimization algorithm that consists of returning the set $\text{SOL} = \text{OPT} \setminus Y$ (that is, the remaining part of the optimum after the deletion of Y). It holds that:

$$w(\text{SOL}) = w(S_1) + w(S_2 \setminus Y) \tag{26}$$

$$w(\text{OPT}') \leqslant w(\text{OPT}' \setminus S_2) + w(S_2 \setminus Y) \tag{27}$$

It also holds that S_1 is an optimal $(k - h')$-colorable subgraph in the induced subgraph $G[V \setminus S_2]$ (otherwise OPT wouldn't be an optimal solution). It also holds that $\text{OPT}' \setminus S_2$ defines a k-colorable subgraph in $G[V \setminus S_2]$. Thus, the $k - h'$ biggest independent sets in $\text{OPT}' \setminus S_2$ have weight at most $w(S_1)$, and at least $\frac{k-h}{k} w(\text{OPT}' \setminus S_2)$. Hence, one verifies that:

$$w(S_1) \geqslant \frac{k - h'}{k} w(\text{OPT}' \setminus S_2) \tag{28}$$

Combining (26), (27) and (28), one finally proves that:

$$\frac{\text{SOL}}{\text{OPT}'} \geqslant \frac{\frac{k-h'}{k} w(\text{OPT}' \setminus S_2) + w(S_2 \setminus Y)}{w(\text{OPT}' \setminus S_2) + w(S_2 \setminus Y)} \geqslant \frac{k - h}{k}$$

that concludes the proof. ∎

Moreover, a simple proof based on the same construction as in Proposition 3 shows that this constant approximation ratio is the best one can obtain by a polynomial algorithm (unless $\mathbf{P} = \mathbf{NP}$).

Proposition 15. *Under deletion of* $h < k$ *vertices,* MAX k-COLORABLE SUBGRAPH *is inapproximable within ratio* $\frac{k-h}{k} + \varepsilon$ *in polynomial time, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. Revisit the proof of Proposition 3: out of an instance H of MAX INDEPENDENT SET, with independence number α, we can build an instance $H_\alpha(V, E)$ of MAX k-COLORABLE SUBGRAPH, such that for any $i \leqslant k$, an optimal i-colorable subgraph in H_α has weight exactly $i\alpha$, and any constant approximation for MAX k-COLORABLE SUBGRAPH is impossible in H_α, unless $\mathbf{P} = \mathbf{NP}$. We build the following reoptimization instance of MAX k-COLORABLE SUBGRAPH, $I_{\alpha,h}$ $(h < k)$:

- The initial graph $G_{\alpha,h}$ is obtained by adding to H_α a clique Y of k vertices, each connected to all vertices of H_α, and each with weight α.
- The perturbed graph $G'_{\alpha,h}$ is obtained by deleting h of the k vertices of Y. Denote by Y' the set of remaining vertices of Y after the deletion ($|Y'| = k - h$).

For reasons already explained in the proof of Proposition 3, it holds that Y can be considered as the initial optimum of the reoptimization instance.

Suppose that, for a given h, there exists an algorithm `A` that computes a $\frac{k-h}{k} + \varepsilon$ approximation for MAX k-COLORABLE SUBGRAPH under deletion of h vertices. And let $S_{\alpha,h}$ denote the solution returned by `A` on the instance $I_{\alpha,h}$, it holds that $w(S_{\alpha,h}) \geqslant (1 + \varepsilon)(k - h)\alpha$, and considering that $w(Y') = (k - h)\alpha$, it holds that $w(S_{\alpha,h}[V]) \geqslant \varepsilon(k - h)\alpha$.

Thus, an algorithm `A` cannot exist for any h, otherwise it would provide a constant approximation for MAX k-COLORABLE SUBGRAPH in the graph $H_\alpha(V, E)$, which is impossible to provide in polynomial time unless $\mathbf{P} = \mathbf{NP}$. ∎

4.4 Restriction to graphs of bounded degree

We will start with the example of MAX INDEPENDENT SET where the underlying technique of algorithm `R1` can be adapted in graphs of bounded degree. We will then try to explain to what extent we can generalize this result.

Proposition 16. *Under-vertex deletion,* MAX INDEPENDENT SET *is approximable within ratio* $1/2$ *in graphs of bounded degree.*

Proof. Let $G(V, E)$ denote an instance of MAX INDEPENDENT SET, with one known optimal solution OPT, and let $G'(V', E') = G[V - \{y\}]$. On this perturbed instance, an optimal solution is denoted by OPT'.

Let us first note that if the deleted vertex y was not in the initial optimum OPT, then this solution remains optimal in the modified graph. Thus, let us suppose that $y \in \mathrm{OPT}$.

In this case, what remains of the optimum is still optimal, but only in the induced subgraph $G'[V' \backslash N(y)]$, where G' is the modified graph, and $N(y)$ is the set of neighbors of y in G. let SOL_1 denote the set $\mathrm{OPT} \setminus \{y\}$, we can directly derive $w(\mathrm{SOL}_1) \geqslant w(\mathrm{OPT}'[V' \setminus N(y)])$. Considering that $|N(y)|$ is bounded by a constant, say Δ, it is possible in $O(2^\Delta)$ to check all possible subsets of $|N(y)|$, and output the maximum weight independent set among them, say SOL_2. Obviously, $w(\mathrm{SOL}_2) \geqslant w(\mathrm{OPT}'[N(y)])$. Returning the best solution among SOL_1 and SOL_2, and taking into account that $w(\mathrm{OPT}') = w(\mathrm{OPT}'[V' \setminus N(y)]) + w(\mathrm{OPT}'[N(y)])$, the result follows. ∎

Note that, provided $\Delta \geqslant 5$, the result beats the classical static approximation ratio of $3/(\Delta + 2)$ [23] for MAX INDEPENDENT SET in graphs of bounded degree.

However, the technique cannot be applied to any hereditary problem. Namely, it can be extended to problems which can be characterized in terms of forbidden subgraphs (and not in terms of forbidden minors), and where the diameter of these forbidden subgraphs is bounded by a constant. Also, all the forbidden subgraphs must be connected components. We denote such problems by MAX H-FREE SUBGRAPH.

Actually, under these conditions, the deletion setting becomes somehow equivalent to the insertion setting:

Proposition 17. *In graphs of degree bounded by Δ, reoptimization of* MAX H-FREE SUBGRAPH *(where each forbidden subgraph has diameter bounded by d) under deletion of a constant number h of vertices is equivalent to reoptimization of the same problem under the insertion of $hd\Delta$ vertices.*

Proof. Consider a reoptimization instance I of MAX H-FREE SUBGRAPH given by an initial graph $G(V, E)$ with degree bounded by Δ, and with a known optimal solution OPT, and a perturbed graph $G'(V', E') = G[V \setminus Y]$, $|Y| = h$.

Recall that all forbidden subgraphs have diameter bounded by a constant d. Let FS (for forbidden subgraph) denote the set of vertices that are reachable from a deleted vertex by a path of order at most d. Obviously $|FS| \leqslant hd\Delta$. It holds that $\mathrm{OPT} \setminus Y$ is an optimal solution on $G'[V' \setminus (FS \setminus \mathrm{OPT})]$.

Indeed, consider a feasible solution S on the graph $G'[V' \setminus (FS \setminus \mathrm{OPT})]$ each vertex of this graph is:

- either not reachable from any deleted vertex by a path of length d, thus it cannot be part of a forbidden subgraph in G along with vertices of $\mathrm{OPT} \cap Y$;

- or in OPT, and considering that OPT is a feasible solution in G, these vertices cannot form a forbidden subgraph in G along with $\mathrm{OPT} \cap Y$.

In all, no vertex in S can form a forbidden subgraph along with $\mathrm{OPT} \cap Y$, so that $S \cup (\mathrm{OPT} \cap Y)$ is necessarily a feasible solution in G. Now, suppose that $w(S) > w(\mathrm{OPT} \setminus Y)$. This induces that $w(S \cup (\mathrm{OPT} \cap Y)) > w(\mathrm{OPT})$, which is impossible considering that $S \cup (\mathrm{OPT} \cap Y)$ is feasible in G. We proved that $\mathrm{OPT} \setminus Y$ is an optimal solution on $G'[V' \setminus (FS \setminus \mathrm{OPT})]$.

Hence, any reoptimization instance I of MAX H-FREE SUBGRAPH under deletion of h vertices can be characterized by:

- a graph $G''(V'', E'') = G'[V' \setminus (FS \setminus \mathrm{OPT})]$ with a known optimal solution $\mathrm{OPT} \setminus Y$;

- a graph $G'(V', E')$ where one wants to optimize the problem. G' contains G'' as a subgraph, and has at most $hd\Delta$ additional vertices with respect to G''.

We just showed that an instance of MAX H-FREE SUBGRAPH, under deletion of h vertices is equivalent to an instance of the problem under insertion of $hd\Delta$ vertices, which concludes the proof. ∎

Recall that, for the case of insertion, another generic algorithm was proposed in [3]. This algorithm, denoted by `R2` uses a polynomial ρ-approximation algorithm for the deterministic problem as subroutine to improve the approximation ratio for the reoptimization version from $\frac{1}{2}$ to $\frac{1}{2-\rho}$. However, considering that most hereditary problems are not constant-approximable in polynomial time (unless $\mathbf{P} = \mathbf{NP}$), `R2` cannot be implemented in general graphs.

Regarding the result of Proposition 17, and considering that MAX INDEPENDENT SET is $3/(\Delta + 2)$ -approximable in graphs of degree bounded by Δ, this generic algorithm can be implemented in the vertex-deletion setting. Indeed, we prove the following result that improves the result of Proposition 16.

Proposition 18. *In graphs of degree bounded by Δ, under deletion of h vertices,* MAX INDEPENDENT SET *is approximable within ratio $\frac{\Delta+2}{2\Delta+1}$ in polynomial time.*

Proof. Let $G(V, E)$ be an instance of MAX INDEPENDENT SET, with one known optimal solution OPT, and let $G'(V', E') = G[V \setminus Y]$, $(|Y| = h)$. On this perturbed instance, an optimal solution is denoted by OPT′. In what follows, $N(S)$ is the set of neighbors of all the vertices in S. Denote by $\mathtt{A}_\rho$ a ρ-approximation algorithm for MAX INDEPENDENT SET in graphs of bounded degree, and $\mathtt{A}_\rho(G)$ a solution returned by $\mathtt{A}_\rho$ on a given graph G. Consider now the following algorithm:

- let $\mathrm{SOL}_1 = \mathrm{OPT} \setminus Y$;
- for each maximal independent set S_i in $N(\mathrm{OPT} \cap Y)$ (they can all be enumerated in $O(2^{h\Delta})$), set $\mathrm{SOL}_{2,i} = S_i \cup \mathtt{A}_\rho(G'[V' \setminus (S_i \cup N(S_i))])$, and $\mathrm{SOL}_2 = \max_i(\mathrm{SOL}_{2,i})$.

For reasons already explained in the proof of Proposition 16, it holds that:

$$w(\mathrm{SOL}_1) \geqslant w(\mathrm{OPT}'[V' \setminus N(\mathrm{OPT} \cap Y)]) \geqslant w(\mathrm{OPT}') - w(\mathrm{OPT}' \cap N(\mathrm{OPT} \cap Y)) \qquad (29)$$

On the other hand, notice that OPT′ can be divided in two parts, $\mathrm{OPT}' \cap N(\mathrm{OPT} \cap Y)$, and $\mathrm{OPT}' \setminus N(\mathrm{OPT} \cap Y)$. Among all sets S_i's computed in the second part of the algorithm, one must have computed the set $S_{i^*} = \mathrm{OPT}' \cap N(\mathrm{OPT} \cap Y)$. Moreover, the second part of the solution, $\mathtt{A}\rho(G'[V' \setminus (S_{i^*} \cup N(S_{i^*})])$ is a ρ approximation on a subgraph where the optimum is $\mathrm{OPT}' \setminus N(\mathrm{OPT} \cap Y)$. In all:

$$\begin{aligned} w(\mathrm{SOL}_2) \geqslant \mathrm{SOL}_{2,i^*} &\geqslant w(\mathrm{OPT}' \cap N(\mathrm{OPT} \cap Y)) + \rho w(\mathrm{OPT}' \setminus N(\mathrm{OPT} \cap Y) \\ &\geqslant \rho w(\mathrm{OPT}') + (1-\rho) w(\mathrm{OPT}' \cap N(\mathrm{OPT} \cap Y)) \end{aligned} \qquad (30)$$

Combining (29) and (30), with coefficients $(1-\rho)$ and 1, one finally proves that:

$$\frac{w(\mathrm{SOL})}{w(\mathrm{OPT}')} \geqslant \frac{1}{2-\rho} \qquad (31)$$

Replacing ρ with $3/(\Delta+2)$ in (31) leads to the claimed result. ∎

5 Reoptimization and BIN PACKING

Given a constant B, a list L of n items $L = (1, 2, \ldots, n$, such that, for any $i = 1, \ldots, n$, its size $a_i \leqslant B$, and n bins each of capacity B, the BIN PACKING problem consists of arranging the items of L in the bins without exceeding their capacity (i.e., the sum of the sizes of the items placed in every bin must not exceed B) and in such a way that a minimum number of bins is used.

We show in this section that the basic technique used before in order to get inapproximability results van be also applied on hereditary problems not necessarily defined on graphs. This is, for instance the case of BIN PACKING. We will prove that this problem is inapproximable within approximation ratio $3/2 - \varepsilon$, for any $\varepsilon > 0$, in the reoptimization setting studied in this paper. For simplicity we consider a non-normalized instance of BIN PACKING where item i, $i \leqslant n$ has integer size a_i and bins have capacity B. We assume also that, for every i, $a_i < B$.

Suppose, ad contrario, that BIN PACKING is approximable within approximation ratio 3/2 under items insertion and consider an instance I of the PARTITION problem, where $n+1$ items $0, 1, \ldots, n$ with sizes $a_0, a_1, \ldots, a_n$ are given such that, for $0 = 1, \ldots, n$, $a_i < B$ and $\sum_{i=0}^{n} a_i = 2B$, and the objective is to find a partition of the items (if any) into two subsets such that the sum of the sizes of their items is equal to B. PARTITION is known to be **NP**-complete.

Assume now that items are ordered in decreasing size order (i.e., $a_0 \geqslant a_1 \geqslant \ldots \geqslant a_n$) and consider the list of items $L = (a_1, \ldots, a_n)$ as instance of BIN PACKING. Obviously, since $a_0 < B$

and $\sum_{i=0}^{n} a_i = 2B$, it holds that $\sum_{i=1}^{n} a_i > B$. Thus, an optimal BIN PACKING-solution for L has value greater than 1. We claim that L has a solution using 2 bins. Indeed, such a solution places the first k items in one bin, where k is the largest index such that $\sum_{i=1}^{k} \leqslant B$, and the rest of the items from $k+1$ to n in a second bin. Let us prove that such a solution is feasible, or equivalently, that $\sum_{i=k+1}^{n} \leqslant B$. Assume, ad contrario, that $\sum_{i=k+1}^{n} > B$, recall that, by the definition of k, $\sum_{i=1}^{k+1} a_i > B$ and observe that $\sum_{i=0}^{k} a_i > \sum_{i=1}^{k+1} a_i > B$. In other words, on the hypothesis that $\sum_{i=k+1}^{n} > B$, we derive that $\sum_{i=0}^{n} a_i = \sum_{i=0}^{k} a_i + \sum_{i=k+1}^{n} > B + B = 2B$, a contradiction. So, the BIN PACKING-instance L has a solution of 2 bins. In all, we can assume that the initial BIN PACKING-instance is L and the optimal solution provided with is as just described.

Assume now that item 0 with size $a_0 \geqslant a_1$ arrives. On the hypothesis of the existence of a polynomial reoptimization algorithm achieving approximation ratio $3/2 - \varepsilon$ for BIN PACKING, if the instance I of PARTITION is a "yes"-instance, then this algorithm would provide a solution with 2 bins, while if I is a "no"-instance, the algorithm would provide a solution with at least 3 bins, deciding so in polynomial time PARTITION.

On the other hand, BIN PACKING is immediately approximable within $3/2$ in the reoptimization setting under consideration. Given an instance L and a solution with k bins, when an element arrives, one can open a new bin in order to place it (after, eventually, a quick check that all the items cannot be placed in the same bin) guaranteeing so an approximation ratio $(k+1)/k \leqslant 3/2$, for $k \geqslant 2$, while the case $k = 1$ is trivially polynomial (just check whether $\sum_{i=1}^{n} a_i \leqslant B$, or not).

Hereditary problems	Approximation ratios	Inapproximability bounds
MAX k-COLORABLE SUBGRAPH (insertion of h vertices)	$\max\left\{\frac{k}{k+h}, \frac{1}{2}\right\}$	$\max\left\{\frac{k}{k+h}, \frac{1}{2}\right\} + \varepsilon$
MAX k-COLORABLE SUBGRAPH (deletion of $h < k$ vertices)	$\frac{k-h}{k}$	$\frac{k-h}{k} + \varepsilon$
MAX k-COLORABLE SUBGRAPH (deletion of $h \geqslant k$ vertices)	$\emptyset$	$n^{-\varepsilon}$
MAX P_k-FREE SUBGRAPH (single vertex insertion)	$\frac{2k}{3k+1}$ or $\frac{2k}{3k+2}$ $(k \leqslant 6)$	$\frac{2k}{3k+1} + \varepsilon$ or $\frac{2k}{3k+2} + \varepsilon$
MAX SPLIT SUBGRAPH (insertion of $h \leqslant 2$ vertices)	$\frac{h+1}{2h+1}$	$\frac{h+1}{2h+1} + \varepsilon$
MAX SPLIT SUBGRAPH (insertion of $h = 3$ vertices)	$\frac{5}{9}$	$\frac{5}{9} + \varepsilon$
MAX SPLIT SUBGRAPH (insertion of $h \geqslant 4$ vertices)	$\frac{1}{2}$	$\frac{1}{2} + \varepsilon$
MAX PLANAR SUBGRAPH (single vertex insertion)	$\frac{2}{3}$	$\frac{4}{5} + \varepsilon$

Table 1: Summary of the results

6 Conclusion

We have discussed the approximability of various hereditary problems in the reoptimization setting where the vertex set is modified. It appears that the initial optimum provides a very useful information when approximating the modified instances, and for most problems discussed, we presented reoptimization algorithms which take advantage of this information in the best possible way, since most approximation ratios that were presented are the best constant ratio achievable in polynomial time (unless $\mathbf{P} = \mathbf{NP}$). Indeed, as Table 1 shows, MAX PLANAR SUBGRAPH is the only among the problems studied where a gap occurs between the approximation ratio and the inapproximability bound (in the general case). Reducing this gap, as well as proving Hypothesis 1 for any k are both subjects of ongoing research.

References

[1] C. Archetti, L. Bertazzi, and M.G. Speranza. Reoptimizing the traveling salesman problem. *Networks*, 42(3):154–159, 2003.

[2] C. Archetti, L. Bertazzi, and M.G. Speranza. Reoptimizing the 0-1 knapsack problem. *Discrete Applied Mathematics*, 158(17):1879–1887, 2010.

[3] G. Ausiello, V. Bonifaci, and B. Escoffier. Complexity and approximation in reoptimization. *CiE 2007: Logic and Computation and Logic in the Real World*, 2007.

[4] G. Ausiello, B. Escoffier, J. Monnot, and V.Th. Paschos. Reoptimization of minimum and maximum traveling salesman's tours. In Lars Arge and Rusins Freivalds, editors, *SWAT*, volume 4059 of *Lecture Notes in Computer Science*, pages 196–207. Springer, 2006.

[5] P. Berman. A d/2 approximation for maximum weight independent set in d-claw free graphs. *Algorithm Theory-SWAT 2000*, pages 31–40, 2000.

[6] P. Berman and T. Fujito. On approximation properties of the independent set problem for degree 3 graphs. *Algorithms and Data Structures*, pages 449–460, 1995.

[7] D. Bilò, H.-J. Böckenhauer, J. Hromkovic, R. Královic, T. Mömke, P. Widmayer, and A. Zych. Reoptimization of steiner trees. In Joachim Gudmundsson, editor, *SWAT*, volume 5124 of *Lecture Notes in Computer Science*, pages 258–269. Springer, 2008.

[8] D. Bilò, P. Widmayer, and A. Zych. Reoptimization of weighted graph and covering problems. In Evripidis Bampis and Martin Skutella, editors, *WAOA*, volume 5426 of *Lecture Notes in Computer Science*, pages 201–213. Springer, 2008.

[9] H.-J. Böckenhauer, L. Forlizzi, J. Hromkovic, J. Kneis, J. Kupke, G. Proietti, and P. Widmayer. On the approximability of tsp on local modifications of optimally solved instances. *Algorithmic Operations Research*, 2(2):83–93, 2007.

[10] H.-J. Böckenhauer, J. Hromkovic, R. Královic, T. Mömke, and P. Rossmanith. Reoptimization of steiner trees: Changing the terminal set. *Theor. Comput. Sci.*, 410(36):3428–3435, 2009.

[11] H.-J. Böckenhauer, J. Hromkovic, T. Mömke, and P. Widmayer. On the hardness of reoptimization. In Viliam Geffert, Juhani Karhumäki, Alberto Bertoni, Bart Preneel, Pavol Návrat, and Mária Bieliková, editors, *SOFSEM*, volume 4910 of *Lecture Notes in Computer Science*, pages 50–65. Springer, 2008.

[12] H.-J. Böckenhauer and D. Komm. Reoptimization of the metric deadline tsp. *J. Discrete Algorithms*, 8(1):87–100, 2010.

[13] R. Boppana and M.M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. *BIT Numerical Mathematics*, 32(2):180–196, 1992.

[14] N. Boria and V.Th. Paschos. Fast reoptimization for the minimum spanning tree problem. *Journal of Discrete Algorithms*, 8(3):296–310, 2010.

[15] J.M. Boyer and W.J. Myrvold. On the cutting edge: Simplified $o(n)$ planarity by edge addition. *Journal of Graph Algorithms and Applications*, 8(3):241–273, 2004.

[16] M. Demange, X. Paradon, and V.Th. Paschos. On-line maximum-order induced hereditary subgraph problems. *International Transactions in Operational Research*, 12(2):185–201, 2005.

[17] M. Demange and V.Th. Paschos. Improved approximations for maximum independent set via approximation chains. *Applied Mathematics Letters*, 10(3):105–110, 1997.

[18] B. Escoffier, M. Milanic, and V.Th. Paschos. Simple and fast reoptimizations for the steiner tree problem. *Algorithmic Operations Research*, 4(2):86–94, 2009.

[19] B. Escoffier and V.Th. Paschos. On-line models and algorithms for max independent set. *RAIRO Operations Research*, 40(2):129–142, 2006.

[20] U. Feige. Approximating maximum clique by removing subgraphs. *SIAM Journal on Discrete Mathematics*, 18:219, 2004.

[21] M.M. Halldórsson. Approximations of weighted independent set and hereditary subset problems. *Journal of Graph Algorithms and Applications*, 4(1):1–16, 2000.

[22] M.M. Halldórsson, K. Iwama, S. Miyazaki, and S. Taketomi. Online independent sets. *Computing and Combinatorics*, pages 202–209, 2000.

[23] M.M. Halldórsson and J. Radhakrishnan. Greed is good: Approximating independent sets in sparse and bounded-degree graphs. *Algorithmica*, 18(1):145–163, 1997.

[24] C. Lund and M. Yannakakis. The approximation of maximum subgraph problems. *Automata, Languages and Programming*, pages 40–51, 1993.

[25] T. Matsui. Approximation algorithms for maximum independent set problems and fractional coloring problems on unit disk graphs. In Jin Akiyama, Mikio Kano, and Masatsugu Urabe, editors, *Discrete and Computational Geometry*, volume 1763 of *Lecture Notes in Computer Science*, pages 194–200. Springer Berlin / Heidelberg, 2000. 10.1007/978-3-540-46515-7-16.

[26] C. Murat and V.Th. Paschos. A priori optimization for the probabilistic maximum independent set problem. *Theor. Comput. Sci.*, 270(1-2):561–590, 2002.

[27] A. Panconesi and Desh Ranjan. Quantifiers and approximation. *Theoretical Computer Science*, 107(1):145 – 163, 1993.

[28] N. Robertson, D.P. Sanders, P. Seymour, and R. Thomas. A new proof of the four-colour theorem. *Electron. Res. Announc. Amer. Math. Soc*, 2(1):17–25, 1996.

[29] N. Robertson and P.D. Seymour. Graph minors. xx. wagner's conjecture. *J. Comb. Theory, Ser. B*, 92(2):325–357, 2004.

[30] A. Wagner. On finite affine line transitive planes. *Mathematische Zeitschrift*, 87(1):1–11, 1965.

[31] D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proc. STOC'06*, pages 681–690, 2006.