Hardness Results and Approximation Algorithms for Discrete Optimization Problems with Conditional and Unconditional Forbidden Vertices

Francois Delbot, Christian Laforest, Raksmey Phan

Hardness Results and Approximation Algorithms for Discrete Optimization Problems with Conditional and Unconditional Forbidden Vertices

Francois Delbot* Christian Laforest† Raksmey Phan‡

April 18, 2016

Abstract

In this paper we study and solve new variants of classical graph problems (vertex cover, dominating set, Steiner tree). We add constraints of incompatibilities between vertices that can be *conditional* or *unconditional*. This capture the impossibility for certains vertices or pairs of vertices of been into a solution.

In the first part, we consider a graph with unconditional forbidden vertices. An instance of the problem is a graph, a set F of *Forbidden vertices* and a set R of *Required vertices*. We prove that constructing a minimal size vertex cover or connected vertex cover or dominating set or Steiner tree, containing all R and no vertex of F can be 2-approximated (when there exists one, that is polynomial to determine). We also show that it is $\mathcal{NP}$*-complete* to determine whether there is an independent dominating set (containing R and no vertex of F).

In the second part, we carry on the conditional case that is expressed by *conflicts* that are a set of pairs of vertices that cannot be both into a solution. An instance is then a graph $\mathcal{G}$ and a set $\mathcal{C}$ of conflicts. We first study the question to know whether there is a vertex cover of $\mathcal{G}$ containing no conflict of $\mathcal{C}$ and if the answer is positive to construct one of minimal size. We reduce that to 2-SAT and we show that the first question can be answered with a polynomial time algorithm. We show that the second problem is $\mathcal{NP}$*-complete* but can be 2-approximated. We also prove that it is $\mathcal{NP}$*-complete* to decide if there exists a connected vertex cover, an (independent) dominating set or a Steiner tree with no conflict of $\mathcal{C}$.

1 Introduction

A traditional discrete graph optimization problem takes as input a graph $\mathcal{G}$ (plus potential supplementary parameters), and the goal consists in general to construct/extract $\mathcal{S}$ a *structure* (depending of the problem) from $\mathcal{G}$, optimizing a given measure (also function of the problem). Several classical problems of this type are the basis of concrete applications. However, in real world, situations are often more complicated. Some elements of $\mathcal{G}$ can be *incompatible* between them and cannot be together in the final solution $\mathcal{S}$. These incompatibilities can come from the nature of the elements

*UPMC - LIP6, Tour 26-00, 4ème étage, 4 place jussieu, 75252 Paris Cedex 05, France

†LIMOS, CNRS UMR 6158 – Université Blaise Pascal, Campus des Cézeaux, 1 rue de la Chebarde, TSA 60125, CS 60026, 63178 Aubière Cedex, France. email: christian.laforest@isima.fr

‡LIMOS, CNRS UMR 6158 – Université Blaise Pascal, Campus des Cézeaux, 1 rue de la Chebarde, TSA 60125, CS 60026, 63178 Aubière Cedex, France. email: phan@isima.fr

and can be due to security reasons, non common interface, capabilities of treatment, etc. This is not the objective of our paper to detect them. We suppose here that they are known and they constitute a part of the input of the problem. The last dimension we must take into account is the fact that two elements can be incompatible, unconditionally or conditionally. The first type of constraints states that some elements cannot be in any solution. We say that they are *forbidden*. The conditional incompatibility between two elements u and v states that *if* one is in $\mathcal{S}$ (for example u) *then* the other one (v in this case) cannot be in $\mathcal{S}$. We say that there is a *conflict* between u and v.

To explain more precisely the problems we study under these two points of view, we first need some notations and definitions.

In this paper, the support will be a *undirected*, *unweighted* graph $\mathcal{G}$. Its set of *vertices* is denoted by $\mathcal{V}$ and its set of edges by $\mathcal{E}$. Two vertices u and v linked by an edge uv are *neighbors* in $\mathcal{G}$. The set of neighbors of u in $\mathcal{G}$ is noted $N_{\mathcal{G}}(u)$ or simply $N(u)$ when there is no ambiguity of the support. Graph $\mathcal{G}$ is *connected* if there is a *path* in $\mathcal{G}$ between each pair of vertices. A *tree* is a connected graph containing no cycle. If F is a subset of vertices, $\mathcal{G}[F]$ is the graph *induced* by F in $\mathcal{G}$: Its set of vertices is F and its edges are only the ones of $\mathcal{G}$ between pairs of vertices of F. An *independent set* F of $\mathcal{G}$ is a set of vertices such that $\mathcal{G}[F]$ contains no edge. In this paper we study variants of the following classical problems.

A *vertex cover* VC is a subset of the vertices ($VC \subseteq \mathcal{V}$) such that each edge uv is *covered* by VC, i.e. has at least one extremity in VC ($u \in VC$ or $v \in VC$ or both). A *connected vertex cover* VC is a vertex cover with the additional property that $\mathcal{G}[VC]$ is connected. A *dominating set* in $\mathcal{G}$ is a subset of vertices $D \subseteq \mathcal{V}$ such that each vertex outside D is neighbor of (is *dominated* by) at least one vertex in D (for all $v \in \mathcal{V} - D$, $N(u) \cap D \neq \emptyset$). An *independent dominating set* I is an independent set of $\mathcal{G}$ ($\mathcal{G}[I]$ has no edge) that is a dominating set of $\mathcal{G}$. Let M be any subset of vertices of $\mathcal{G}$ ($M \subseteq \mathcal{V}$); a *Steiner Tree* for the instance $(\mathcal{G}, M)$ is a *tree* $T = (\mathcal{V}_T, \mathcal{E}_T)$ covering M ($M \subseteq \mathcal{V}_T \subseteq \mathcal{V}$) in $\mathcal{G}$ ($\mathcal{E}_T \subseteq \mathcal{E}$).

The *size* of a *solution* $\mathcal{S}$ to these problems (namely a vertex cover (connected or not) or a dominating set (independent or not)), is the number of its vertices, noted $|\mathcal{S}|$. The size, or weight, of a Steiner tree is its number of edges (or equivalently its number of vertices minus one, by a well-known property of trees).

All these problems are classical, their optimization versions are $\mathcal{NP}$*-hard* (see [4]) and there are approximation algorithms to solve them (some with constant approximation ratios).

As mentioned at the beginning of the paper, we study variations of these 5 problems, taking into account the potential incompatibilities between elements, that are of two kinds.

- In *unconditional* compatibilities, study in Section 2, we are given two additional sets of vertices, namely set F of *forbidden* vertices that cannot be in any solution. We also consider a set R of *required* vertices that *must* be in any solution. Of course we suppose that $R \cap F = \emptyset$.

- *Conditional* incompatibilities are given, in Section 3, as a set of pairs of vertices that cannot be *together* in any solution. This is called a *conflict*. If there is a conflict between u and v then both of them cannot be in a solution (at most one of them can be).

With these new constraints, the existence of a solution, regardless of its size, is not guaranteed. Thus, we proceed in two steps. First, we study the problem of determining the existence of a solution. If this problem is polynomial then, in a second stage, we propose approximation algorithms to solve the optimization version and we prove their approximation ratios.

Several works have been done on similar contexts. Recently several papers have been published concerning the construction of structures in graphs under constrains like *conflicts* of *edges*: If two edges e and e' are in conflict they cannot be part of the same structure (at most one can be into the structure, not both). This has been investigated for paths, trees, Hamiltonian paths or cycles in [2, 7, 8, 11, 12, 15]. The same kind of study has been done when the conflicts concern pairs of vertices (see [9] for example). All these constrains are conditional: If an edge (or a vertex) is in the structure then the other one cannot be part of it. Concrete applications may concern incompatible devices or links in a network for example.

2 Optimization Problems in Graphs With Unconditional Forbidden Vertices

This section is devoted to study variants of the 5 classical discrete optimization problems mentioned in the introduction, under the *unconditional* constraints. In addition to the support graph $\mathcal{G}$, two sets F (forbidden vertices) and R (require vertices) are given into the instance that is now of the form $(\mathcal{G}, F, R)$. The existence of a solution including all the vertices of R and containing no vertex of F is studied. If this can be determined in polynomial time, then we solve the optimization problem by proposing an approximation algorithm.

2.1 Vertex Cover in $(\mathcal{G}, F, R)$

Given $(\mathcal{G}, F, R)$ a *Vertex Cover with Forbidden and Required Vertices* (*VCwFaRV*) $\mathcal{S}$ is a vertex cover of $\mathcal{G}$ (for each edge $e = uv$ of $\mathcal{G}$, u or v (or both) is in $\mathcal{S}$) such that $F \cap \mathcal{S} = \emptyset$ and $R \subseteq \mathcal{S}$. We first make the following remark, easy to prove.

Proposition 1 $(\mathcal{G}, F, R)$ *contains a* VCwFaRV *if and only if F is an independent set of $\mathcal{G}$ and $F \cap R = \emptyset$.*

This remark proves that deciding whether there is a *VCwFaRV* in $(\mathcal{G}, F, R)$ is polynomial. Minimizing the size of a *VCwFaRV* is hard, since the very particular case $R = F = \emptyset$ is the classical $\mathcal{NP}$*-complete* vertex cover problem (see [4]). We propose an approximation algorithm for the minimal size *VCwFaRV* problem. Our algorithm uses as a subroutine any 2-approximation algorithm, noted A, for the optimal vertex cover problem (some can be found in [16], see [3] for a recent new one).

Input: Any instance $(\mathcal{G}, F, R)$ such that $F \cap R = \emptyset$ and F an independent set of $\mathcal{G}$.
(1) $N(F) = \{v : uv \in \mathcal{E}, u \in F\}$ (the set of neighbors of the vertices of F in $\mathcal{G}$).
(2) Let $\mathcal{V}' = \mathcal{V} - (F \cup R \cup N(F))$ and $\mathcal{G}' = \mathcal{G}[\mathcal{V}']$ be the graph induced by $\mathcal{V}'$ in $\mathcal{G}$.
(3) Apply algorithm A on $\mathcal{G}$', that returns $\mathcal{S}'$.
(4) Return $\mathcal{S} = \mathcal{S}' \cup R \cup N(F)$.

Theorem 2.1 *Our algorithm is a 2-approximation algorithm for the minimal size* VCwFaRV *problem.*

`Preuve.` Our algorithm is polynomial (subroutine A is polynomial). By construction, the set $\mathcal{S}$ returned by our algorithm contains all vertices of R and no vertices of F. Let $e = uv$ be any edge of $\mathcal{G}$. Let us show that e is covered by $\mathcal{S}$. If u and v are in $\mathcal{V}$' then $\mathcal{S}$' covers e since algorithm A returns a vertex cover of $\mathcal{G}$'. Now, if u or v is in R or in $N(F)$, e is covered by $\mathcal{S}$. The last case is when u (or v) is in F. Since F is an independent set, if $u \in F$ then $v \notin F$. But in this case, $v \in N(F)$. Edge e is then covered by $\mathcal{S}$. Our algorithm returns a *VCwFaRV* of instance $(\mathcal{G}, F, R)$. We analyze now its size $|\mathcal{S}|$.

Let $\mathcal{S}^*$ be a minimal size *VCwFaRV* of $(\mathcal{G}, F, R)$ and $OPT = |\mathcal{S}^*|$ be its size. Let $T = \mathcal{S}^* - (R \cup N(F))$. First note that $OPT = |T| + |R \cup N(F)|$ since $R \cup N(F) \subseteq \mathcal{S}^*$ ($N(F) \subseteq \mathcal{S}^*$ because edges incident to F must be covered). T is clearly a vertex cover of $\mathcal{G}$'. We prove by contradiction that T is an optimal vertex cover of $\mathcal{G}' = \mathcal{G}[\mathcal{V}']$.

Suppose the opposite; this means that there exists a vertex cover T_2 of $\mathcal{G}'$ with $|T_2| < |T|$; Hence $\mathcal{S}_2 = T_2 \cup R \cup N(F)$ is a *VCwFaRV* of size $|\mathcal{S}_2| \leq |T_2| + |R \cup N(F)| < |T| + |R \cup N(F)| = |\mathcal{S}^*| = OPT$. This is in contradiction with the optimality of $\mathcal{S}^*$.

As $\mathcal{S}'$ is a 2-approximation of an optimal vertex cover of $\mathcal{G}'$ (by property of algorithm A) we get: $|\mathcal{S}'| \leq 2|T|$. Moreover, as $OPT = |T| + |R \cup N(F)|$ we have: $|T| \leq OPT$. Combining all these points we get:

$$|\mathcal{S}| \leq |\mathcal{S}'| + |R \cup N(F)| \leq 2|T| + |R \cup N(F)| \leq OPT + |T| + |R \cup N(F)| = 2OPT$$

□

2.2 Connected Vertex Cover in $(\mathcal{G}, F, R)$

Given $(\mathcal{G}, F, R)$, a *Connected Vertex Cover with Forbidden and Required Vertices* (*CVCwFaRV*) $\mathcal{S}$ is a vertex cover of $\mathcal{G}$ (each edge uv has at least one extremity in $\mathcal{S}$) such that $\mathcal{G}[\mathcal{S}]$ is connected, $F \cap \mathcal{S} = \emptyset$ and $R \subseteq \mathcal{S}$. We suppose in this Section that $\mathcal{G}$ is connected.

Proposition 2 *Let $\mathcal{G}$ be a connected graph. $(\mathcal{G}, F, R)$ contains a* CVCwFaRV *if and only if F is an independent set of $\mathcal{G}$, $F \cap R = \emptyset$ and $\mathcal{G}[\mathcal{V} - F]$ is connected.*

`Preuve.` If $(\mathcal{G}, F, R)$ contains a *CVCwFaRV* noted $\mathcal{S}$, this means that $F \cap R = \emptyset$ and F is an independent of $\mathcal{G}$ (otherwise edges of $\mathcal{G}[F]$ could not be covered). Moreover, as $\mathcal{S}$ is a vertex cover of $\mathcal{G}$, any edge uv of $\mathcal{G}$ has at least one extremity in $\mathcal{S}$. Let us consider any vertex $u \in \mathcal{V} - F$: u is in $\mathcal{S}$ or has a neighbor in $\mathcal{S}$ (since $\mathcal{G}$ is connected, u has at least one neighbor) or both. Thus for any $u, v \in \mathcal{V} - F$ there is a path between u and v in $\mathcal{G}[\mathcal{V} - F]$ (through the connected graph $\mathcal{G}[\mathcal{S}]$) which is then connected.

Now suppose that $\mathcal{G}[\mathcal{V} - F]$ is connected, that $F \cap R \neq \emptyset$ and that F is an independent set of $\mathcal{G}$. This means that $\mathcal{S} = \mathcal{V} - F$ contains no vertex of F and contains all vertices of R. Moreover, let uv be any edge of $\mathcal{G}$. As F is an independent set of $\mathcal{G}$, $u \in \mathcal{S}$ or $v \in \mathcal{S}$ (or both). This means that $\mathcal{S}$ is a vertex cover of $\mathcal{G}$. In conclusion, $\mathcal{S}$ is a *CVCwFaRV* of $(\mathcal{G}, F, R)$. □

The previous result shows that deciding whether there exists a *CVCwFaRV* of $(\mathcal{G}, F, R)$ is polynomial and, if it is the case, to construct one (namely $\mathcal{V} - F$). If $F = R = \emptyset$, this is the classical connected vertex cover $\mathcal{NP}$*-complete* problem (see [4]). We propose now an approximation algorithm for the minimal size *CVCwFaRV* problem. Our algorithm uses as subroutine any

2-approximation algorithm, noted A, for the minimal size connected vertex cover problem (see [14] for example).

Input: Any instance $(\mathcal{G}, F, R)$ where $F \cap R = \emptyset$, F is an independent set of $\mathcal{G}$ and $\mathcal{G}[\mathcal{V} - F]$ is connected.
(1) Let $R' = R \cup N(F)$ where $N(F) = \{v : uv \in \mathcal{E}, u \in F\}$.
(2) Construct $\mathcal{G}_+ = (\mathcal{V}_+, \mathcal{E}_+)$ the graph $\mathcal{G}[\mathcal{V} - F]$ in which for each vertex r of R' is added a new proper neighbor r_+ (of degree 1, that is a *leaf* of $\mathcal{G}$), and the associated new edge rr_+.
(3) Apply algorithm A on $\mathcal{G}_+ = (\mathcal{V}_+, \mathcal{E}_+)$ that constructs a connected vertex cover of $\mathcal{G}_+$, noted $\mathcal{S}_+$.
(4) Construction of $\mathcal{S}$. For each new edge rr_+ ($\mathcal{S}_+$ necessarily contains r, r_+ or both), put r in $\mathcal{S}$. Put in $\mathcal{S}$ all the other vertices of $\mathcal{S}_+$ that are not r_+-type vertices.
(5) Return $\mathcal{S}$.

Theorem 2.2 *Our algorithm is a 2-approximation algorithm for the minimal size* CVCwFaRV *problem.*

`Preuve.` Our algorithm is polynomial (algorithm A is polynomial). By hypothesis graph $\mathcal{G}[\mathcal{V} - F]$ is connected. Adding the new vertices r_+ and new edges rr_+ into it maintains the connectivity, hence $\mathcal{G}_+$ is connected. Algorithm A constructs a connected vertex cover $\mathcal{S}_+$ of $\mathcal{G}_+$. After potential modifications at line (4), $\mathcal{S}$ is always a connected vertex cover of $\mathcal{G}_+$, including no r_+-type vertex. $\mathcal{S}$ is then a connected vertex cover of $\mathcal{G}[\mathcal{V} - F]$, $\mathcal{G}[\mathcal{S}]$ is connected and contains no vertices of F. We just need now to show that edges with one extremity in F are also covered to show that $\mathcal{S}$ is a vertex cover of $\mathcal{G}$. Let ur be such an edge, with $u \in F$ and $r \notin F$, that is $r \in N(F)$. By construction of $\mathcal{G}_+$ and by operation of line (4), r is included into $\mathcal{S}$, and edge ur is then covered by $\mathcal{S}$. Again, by construction, any $r \in R$ is also in $\mathcal{S}$ (see line (4)) i.e. $R \subseteq \mathcal{S}$. All these points show that $\mathcal{S}$ is a *CVCwFaRV* of instance $(\mathcal{G}, F, R)$. We must now analyse its size.

Let us study the approximation ratio. Let OPT_+ be the size of an optimal connected vertex cover of $\mathcal{G}_+$. As algorithm A is a 2-approximation algorithm, $|\mathcal{S}_+| \leq 2OPT_+$. By construction, $|\mathcal{S}| \leq |\mathcal{S}_+|$ since for any edge rr_+, $\mathcal{S}$ contains one vertex (r) while $\mathcal{S}_+$ contains at least one (potentially both).

Let $\mathcal{S}^*$ be an optimal *CVCwFaRV* of $(\mathcal{G}, F, R)$ and $OPT = |\mathcal{S}^*|$ be its size. By definition, $\mathcal{S}^*$ contains R, $\mathcal{S}^* \cap F = \emptyset$ and covers every edges of $\mathcal{G}$ thus also contains $N(F)$. Moreover $\mathcal{G}[\mathcal{S}^*]$ is connected. Hence, $\mathcal{S}^*$ contains R', is a connected vertex cover of $\mathcal{G}_+$ and its size is then larger than the optimal one: $OPT_+ \leq OPT$. Combining all these inequalities we get:

$$|\mathcal{S}| \leq |\mathcal{S}_+| \leq 2OPT_+ \leq 2OPT$$

□

2.3 Dominating Set in $(\mathcal{G}, F, R)$

Given $(\mathcal{G}, F, R)$, a *Dominating set with Forbidden and Required Vertices* (*DwFaRV*) $\mathcal{S}$ is a dominating set of $\mathcal{G}$ ($\forall u \notin \mathcal{S}$, $N(u) \cap \mathcal{S} \neq \emptyset$), $F \cap \mathcal{S} = \emptyset$ and $R \subseteq \mathcal{S}$.

Proposition 3 *$(\mathcal{G}, F, R)$ contains a DwFaRV if and only if $F \cap R = \emptyset$ and $\forall u \in F$, $N(u) \cap (\mathcal{V} - F) \neq \emptyset$.*

`Preuve.` Suppose that $\mathcal{S}$ is any $DwFaRV$ of the instance $(\mathcal{G}, F, R)$. Of course $F \cap R = \emptyset$. $\mathcal{S}$ is a dominating set, containing no vertex of F. Thus, any vertex $u \in F$ is dominated by at least one neighbor outside F in $\mathcal{S} \subseteq \mathcal{V} - F$.
Now suppose $F \cap R = \emptyset$ and $\forall u \in F,\ N(u) \cap (\mathcal{V} - F) \neq \emptyset$. Then $\mathcal{S} = (\mathcal{V} - F)$ is a dominating set of $\mathcal{G}$ containing R and no vertex of F. This is a $DwFaRV$. $\square$

This proposition shows that deciding whether the instance $(\mathcal{G}, F, R)$ contains a $DwFaRV$ is polynomial. When $R = F = \emptyset$, a $DwFaRV$ is a classical dominating problem that is $\mathcal{NP}$-*complete* (see [4]). We propose now an approximation algorithm for the minimal size $DwFaRV$ problem. To describe it, we first need to remind the *set cover problem*, another classical problem. It takes a "background" set $\mathcal{X}$ and a family $\mathcal{FAM}$ of subsets of $\mathcal{X}$, covering $\mathcal{X}$ (i.e. $\bigcup_{H \in \mathcal{FAM}} H = \mathcal{X}$). A *set cover* of $(\mathcal{X}, \mathcal{FAM})$ is a subfamily $\mathcal{FAM}$' of $\mathcal{FAM}$ ($\mathcal{FAM}' \subseteq \mathcal{FAM}$), covering $\mathcal{X}$ ($\bigcup_{H \in \mathcal{FAM}'} H = \mathcal{X}$). The size of $\mathcal{FAM}$', noted $|\mathcal{FAM}'|$, is the number of sets it contains. Constructing a set cover of minimal size is an $\mathcal{NP}$-*hard* problem (see [4]). It cannot be approximated by a constant. But there is a well-known greedy algorithm, noted A, that constructs a set cover of $(\mathcal{X}, \mathcal{FAM})$ whose size is at most $1 + \ln(|\mathcal{X}|)$ times the size of an optimal one (see [6]). We use A as a subroutine for our algorithm.

Input: Any instance $(\mathcal{G}, F, R)$ with $F \cap R = \emptyset$ and $\forall u \in F,\ N(u) \cap (\mathcal{V} - F) \neq \emptyset$.
(1) For each vertex u of $\mathcal{G}$, construct set $V(u) = N(u) \cup \{u\}$ (u and all its neighbors). We call u the *center* of $V(u)$.
(2) Apply algorithm A on the set cover instance $(V, \mathcal{FAM})$ with $\mathcal{FAM} = \{V(u) : u \in \mathcal{V} - F\}$ ($\mathcal{FAM}$ contains all the $V(u)$, except the ones associated with vertices of F). Note $\mathcal{FAM}$' the solution returned by A.
(3) $\mathcal{S}' = \{u : V(u) \in \mathcal{FAM}'\}$. $\mathcal{S}$' is composed of the centers u of sets $V(u)$ selected by A.
(4) Return $\mathcal{S} = \mathcal{S}' \cup R$.

Theorem 2.3 *Our algorithm is a* $(2 + \ln(|\mathcal{V}|))$*-approximation algorithm for the minimal size* $DwFaRV$ *problem.*

`Preuve.` Our algorithm is polynomial (A is polynomial). Family $\mathcal{FAM}$ contains no set $V(u)$ with $u \in F$, hence $\mathcal{S}$ contains no vertex of F. By hypothesis $\forall u \in F,\ N(u) \cap (\mathcal{V} - F) \neq \emptyset$, hence family $\mathcal{FAM}$ covers $\mathcal{V}$ and the set cover instance $(\mathcal{V}, \mathcal{FAM})$ has a solution. By construction, $\mathcal{S}$ contains R (line (4)). Let us show now that $\mathcal{S}$ is a dominating set of $\mathcal{G}$. In fact we show that $\mathcal{S}$' (constructed in line (3)) is a dominating set of $\mathcal{G}$ and, as $\mathcal{S}' \subseteq \mathcal{S}$, this is sufficient.

For that purpose we describe now a bijection, preserving the size, between sets cover of $(\mathcal{V}, \mathcal{FAM})$ (with $\mathcal{FAM} = \{V(u) : u \in \mathcal{V} - F\}$) and dominating sets of $\mathcal{G}$, containing no vertex of F.

Let C be any set cover of $(\mathcal{V}, \mathcal{FAM})$. From C construct $S(C)$, the set of centers of sets composing the cover C: $S(C) = \{u : V(u) \in C\}$. With this transformation, any set cover C corresponds to a unique $S(C)$ and, by construction, $|C| = |S(C)|$. Note that $S(C) \cap F = \emptyset$ since $\mathcal{FAM}$ contains no $V(u)$ with $u \in F$. We need to show now that $S(C)$ is a dominating set of $\mathcal{G}$. Let $u \notin S(C)$ be any vertex. As u is an element of $\mathcal{V}$, covered by at least a set $V(v)$ of C ($u \in V(v)$), then $v \in S(C)$ and v, neighbor of u, dominates u.

Consider now the reverse association. Let S be any dominating set of $\mathcal{G}$, containing no vertex of F. From S construct $C(S) = \{V(u) : u \in S\} \subseteq \mathcal{FAM}$. When S is given, $C(S)$ is unique and $|C(S)| = |S|$. Moreover $C(S)$ is a set cover of $\mathcal{V}$ (proof similar to the previous one). These two reverse transformations define the bijection.

Now, let C^* be an optimal size set cover of $(\mathcal{V}, \mathcal{FAM})$. By the bijection, the corresponding dominating set $S(C^*)$ is then a minimal size dominating set of $\mathcal{G}$, containing no vertex of F and $|S(C^*)| = |C^*|$. By property of subroutine A, $|\mathcal{FAM}'| \leq (1+\ln(|\mathcal{V}|))|C^*|$. Our algorithm constructs $\mathcal{S}$' from $\mathcal{FAM}$', exactly as in the bijection. $\mathcal{S}$' is then a dominating set of $\mathcal{G}$, with no vertex of F and $|\mathcal{S}'| = |\mathcal{FAM}'|$. Combining all these elements leads to:

$$|\mathcal{S}'| = |\mathcal{FAM}'| \leq (1 + \ln(|\mathcal{V}|))|C^*| \leq (1 + \ln(|\mathcal{V}|))|S(C^*)|$$

Now let OPT be the size of a minimal size $DwFaRV$ of the initial instance $(\mathcal{G}, F, R)$. As OPT is the size of a dominating set of $\mathcal{G}$, with no vertex of F and all vertices of R, its size is larger that the one of $S(C^*)$ (an optimal size dominating set of $\mathcal{G}$ with the only constraint of having no vertex of F). Hence $|S(C^*)| \leq OPT$. This new inequality implies:

$$|\mathcal{S}'| \leq (1 + \ln(|\mathcal{V}|))OPT$$

The last part of the analysis concerns operation of line (4): $\mathcal{S} = \mathcal{S}' \cup R$, thus $|\mathcal{S}| \leq |\mathcal{S}'| + |R|$. As R must be included in any solution, $|R| \leq OPT$. At the end, we get the expected result $|\mathcal{S}| \leq (2 + \ln(|\mathcal{V}|))OPT$. □

2.4 Independent Dominating Set in $(\mathcal{G}, F, R)$

An *Independent Dominating Set with Forbidden and Required Vertices* (*IDwFaRV*) $\mathcal{S}$ in $(\mathcal{G}, F, R)$ is an independent ($\mathcal{S}$ is an independent set of $\mathcal{G}$), dominating set of $\mathcal{G}$ (each $u \notin \mathcal{S}$ has at least one neighbor in $\mathcal{S}$), containing all vertices of R ($R \subseteq \mathcal{S}$) and no vertices of F ($F \cap \mathcal{S} = \emptyset$). When $R = F = \emptyset$ this is the classical independent dominating set problem. In this case, it is well known that any maximal independent set is also an independent dominating set. Adding the constraints on R and F leads to a $\mathcal{NP}$*-complete* problem as stated in the next Theorem.

Theorem 2.4 *Given* $(\mathcal{G}, F, R)$*, deciding whether there exists an* IDwFaRV *is* $\mathcal{NP}$-complete*, even if* $R = \emptyset$*.*

`Preuve.` The problem is clearly in $\mathcal{NP}$. We reduce it to the *X3C* (Exact Cover by 3 Sets) $\mathcal{NP}$*-complete* problem (see [4]) that we remind now. Let $\mathcal{X} = \{u_1, u_2, \ldots, u_{3q}\}$ a set of $3q$ *elements* and $\mathcal{H}$ a family of k sets $C_i \subseteq \mathcal{X}$ such that $|C_i| = 3$ ($i \in \{1, \ldots, k\}$) and $\cup_{i=1}^{k} C_i = \mathcal{X}$. Given the instance $(\mathcal{X}, \mathcal{H})$, the *X3C* problem is to determine whether there exists an *exact cover* $\mathcal{S}_{\mathcal{H}} \subseteq \mathcal{H}$ of $\mathcal{X}$: $\forall C_i \in \mathcal{S}_{\mathcal{H}}, \forall C_j \in \mathcal{S}_{\mathcal{H}} - \{C_i\}, C_i \cap C_j = \emptyset$ and $\displaystyle\bigcup_{C_i \in \mathcal{S}_{\mathcal{H}}} C_i = \mathcal{X}$.

Let $(\mathcal{X}, \mathcal{H})$ be any *X3C* instance. We construct an instance $(\mathcal{G}, F, R)$ as follows. Each $u_j \in \mathcal{X}$ becomes a vertex (also noted u_j) of $\mathcal{G}$. Each $C_i \in \mathcal{H}$ becomes a vertex (also noted C_i) of $\mathcal{G}$. In $\mathcal{G}$ each vertex C_i is connected to the 3 vertices u_a, u_b, u_c if, in $\mathcal{H}$, $C_i = \{u_a, u_b, u_c\}$. Two distinct vertices C_i and C_j are connected in $\mathcal{G}$ if, in $\mathcal{H}$, the two sets are non-disjoint: $C_i \cap C_j \neq \emptyset$. The set F of forbidden vertices is $\mathcal{X}$: $F = \mathcal{X}$. The set R of required vertices is empty: $R = \emptyset$. This construction can be done in polynomial time. For any *X3C* instance $(\mathcal{X}, \mathcal{H})$ we note $(\mathcal{G}, F, R)$ the associated instance of our problem.

Figure 1 displays an example to get *IDwFaRV* from a solution of a *X3C* instance: $\mathcal{X} = \{u_1, \ldots, u_9\}$ and $C_1 = \{u_1, u_2, u_3\}$, $C_2 = \{u_3, u_4, u_5\}$, $C_3 = \{u_5, u_6, u_7\}$ and $C_4 = \{u_4, u_8, u_9\}$.

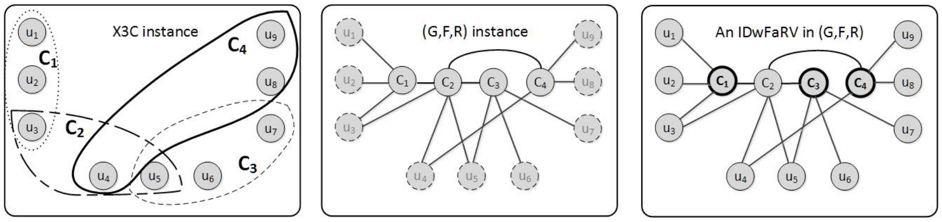

Figure 1: Solution of *IDwFaRV* from *X3C* instance.

Suppose first that there is a solution $\mathcal{S}_{\mathcal{H}}$ for the *X3C* problem. Let $\mathcal{S}$ be the set composed by all the corresponding vertices of $\mathcal{S}_{\mathcal{H}}$. $\mathcal{S}$ is an independent set of $\mathcal{G}$: Indeed, as $\mathcal{S}_{\mathcal{H}}$ is an exact cover of $\mathcal{X}$, all the sets $C_i \in \mathcal{S}_{\mathcal{H}}$ are pairwise disjoint and then the corresponding vertices of $\mathcal{S}$ are non connected in $\mathcal{G}$. $\mathcal{S}$ is also a dominating set of $\mathcal{G}$ because each vertex u_i is dominated by exactly one $C_j \in \mathcal{S}_{\mathcal{H}}$ (the one containing it) and each C_j that is not in $\mathcal{S}$ is dominated by at least a vertex in $\mathcal{S}$; Indeed, as $\mathcal{S}_{\mathcal{H}}$ is a cover of $\mathcal{X}$, the 3 elements of set C_j are covered by sets of $\mathcal{S}_{\mathcal{H}}$ meaning that there is $C_i \in \mathcal{S}_{\mathcal{H}}$ such that $C_i \cap C_j \neq \emptyset$; vertex $C_j \notin \mathcal{S}$ is then dominated by vertex $C_i \in \mathcal{S}$ in $\mathcal{G}$ by the edge C_iC_j. To finish, as $\mathcal{S}$ contains no forbidden vertices (because $F = \mathcal{X}$), $\mathcal{S}$ is a *IDwFaRV* of $(\mathcal{G}, F, R)$.

Suppose now that there is an *IDwFaRV* noted $\mathcal{S}$, of $(\mathcal{G}, F, R)$. In this case, $\mathcal{S}$ contains no vertex of $\mathcal{X}$ (because these vertices are forbidden, $F = \mathcal{X}$). Hence $\mathcal{S}$ must contain some vertices C_i. Note $\mathcal{S}_{\mathcal{H}}$ the family of sets corresponding to the vertices of $\mathcal{S}$. As $\mathcal{S}$ is a dominating set of $\mathcal{G}$, $\cup_{C_i \in \mathcal{S}_{\mathcal{H}}} C_i = \mathcal{X}$. Moreover, as $\mathcal{S}$ is an independent set of $\mathcal{G}$, for all $C_i \in \mathcal{S}_{\mathcal{H}}$ and all $C_j \in \mathcal{S}_{\mathcal{H}}$, there is no edge connecting them, i.e. $C_i \cap C_j = \emptyset$. Thus $\mathcal{S}_{\mathcal{H}}$ is a solution for the *X3C* problem. □

This problem is harder than the others, since even the question of the existence of a solution (regardless of its size) is $\mathcal{NP}$*-complete*.

2.5 Steiner Tree in $(\mathcal{G}, F, R)$

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be any graph and M be any subset of vertices of $\mathcal{G}$ ($M \subseteq \mathcal{V}$). A *Steiner Tree with Forbidden and Required Vertices* (*STwFaRV*) for the instance $(\mathcal{G}, M, F, R)$ is a *tree* $T = (\mathcal{V}_T, \mathcal{E}_T)$, covering M ($M \subseteq \mathcal{V}_T \subseteq \mathcal{V}$) in $\mathcal{G}$ ($\mathcal{E}_T \subseteq \mathcal{E}$), containing no vertex of F ($F \cap \mathcal{V}_T = \emptyset$) and containing all vertices of R ($R \subseteq \mathcal{V}_T$). Note that when $F = R = \emptyset$ this tree is a classical Steiner tree of M in $\mathcal{G}$. Given an instance $(\mathcal{G}, M, F, R)$, the first question is to know when such a tree exists. The next result, easy to prove, gives the condition that can be verified in polynomial time.

Proposition 4 *$(\mathcal{G}, M, F, R)$ contains a* STwFaRV *if and only if $F \cap R = \emptyset$ and all the vertices of $M \cup R$ are in the same connected component of graph $\mathcal{G}[\mathcal{V} - F]$.*

Now, for instances satisfying conditions of Proposition 4, the problem is to minimize the size of the solution (i.e. the number of edges in the tree). But when $F = R = \emptyset$ this problem is equivalent to the minimal size Steiner tree problem which is $\mathcal{NP}$*-hard* (see [4]). In the following we propose an approximation algorithm. To describe it we will use, as a subroutine, any polynomial time ρ-approximation algorithm, noted A, for the classical minimal size Steiner tree (for example A can be the algorithm described in [13], with $\rho = 1 + \frac{\ln 3}{2}$).

Input: Any instance $(\mathcal{G}, M, F, R)$ such that $F \cap R = \emptyset$ and all the vertices of $M \cup R$ are in the same connected component of graph $\mathcal{G}[\mathcal{V} - F]$.
(1) Extract $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ the connected component of $\mathcal{G}[\mathcal{V} - F]$ that contains $M' = M \cup R$.
(2) Apply algorithm A on the Steiner tree instance $(\mathcal{G}', M')$.
(3) Return $T = (\mathcal{V}_T, \mathcal{E}_T)$ the tree constructed in step (2).

Theorem 2.5 *The previous algorithm is a ρ-approximation for the minimal size* STwFaRV *problem.*

`Preuve.` First note that our algorithm is polynomial (the subroutine A is polynomial). As $\mathcal{G}'$ is connected and contains $M \cup R$, algorithm A construct a Steiner tree of $M' = M \cup R$ in $\mathcal{G}'$. Hence, the constructed tree T contains all the vertices of M and R. Moreover, by construction, $\mathcal{G}'$, and then also T, contain no vertex of F. Tree T is then a *STwFaRV* of instance $(\mathcal{G}, M, F, R)$. Let us analyse its size, noted $s(T) = |\mathcal{E}_T|$, the number of its edges.

Let $T^* = (\mathcal{V}^*, \mathcal{E}^*)$ be a minimal size *STwFaRV* of $(\mathcal{G}, M, F, R)$; $s(T^*) = |\mathcal{E}^*|$ is its size. Let $\mathcal{S} = (\mathcal{V}_\mathcal{S}, \mathcal{E}_\mathcal{S})$ be a minimal size Steiner tree of $(\mathcal{G}', M')$; $s(\mathcal{S}) = |\mathcal{E}_\mathcal{S}|$ is its size. By definition of A we have $s(T) \leq \rho \cdot s(\mathcal{S})$. Moreover, note that T^* is a tree containing all vertices of M'. This means that it is a Steiner tree of $(\mathcal{G}', M')$. Its size is then larger than the one of $\mathcal{S}$, a minimal size Steiner tree of $(\mathcal{G}', M')$. Hence $s(\mathcal{S}) \leq s(T^*)$. Combining the two inequalities leads to $s(T) \leq \rho \cdot s(T^*)$ as expected. □

3 Optimization Problems in Graphs With Conditional Forbidden Vertices (Conflicts)

In Section 2 we studied unconditional constraints for 5 optimization problems. Here we turn our attention to conditional incompatibilities named *conflicts*. A conflict is a *pair of vertices* $\{u, v\}$ of $\mathcal{G}$ and we note $\mathcal{C}$ the set of conflicts. Given such an instance $(\mathcal{G}, \mathcal{C})$ the goal is to extract a solution from $\mathcal{G}$ (a vertex cover (connected or not) or a dominating set (independent or not) or a Steiner tree of a given set M of vertices) containing no conflict of $\mathcal{C}$. It is easy to see that it is not always possible. In Section 3.1 we show that constructing a vertex cover with no conflict in $(\mathcal{G}, \mathcal{C})$ when there is one is polynomial and we propose a 2-approximation algorithm to solve the problem of minimization of its size. In other sections we prove that it is *$\mathcal{NP}$-complete* to determine whether there exists a solution for the *connected vertex cover with no conflict*, the *dominating set with no conflict*, the *independent dominating set with no conflict* and the *Steiner tree with no conflict*.

3.1 Vertex Cover With no conflict

This section is devoted to the variant of the vertex cover in which any pair of vertices cannot simultaneously be in a solution. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be any graph and $\mathcal{C}$ be a set of *conflicts* of $\mathcal{G}$. Given $(\mathcal{G}, \mathcal{C})$, a *Vertex Cover with no conflict* (*VCwnC*) is a vertex cover $\mathcal{S}$ of $\mathcal{G}$ containing no conflict (for all u and v in $\mathcal{S}$, $\{u, v\} \notin \mathcal{C}$). Note that a *VCwnC* does not always exist. An *optimal VCwnC* $\mathcal{S}^*$ is a *VCwnC* of *minimal* size. Given $\mathcal{G}$ and $\mathcal{C}$, constructing an optimal *VCwnC* is an *$\mathcal{NP}$-hard* problem since even in the particular case of an empty set $\mathcal{C}$ of conflicts it is equivalent to construct an optimal vertex cover in $\mathcal{G}$.

In the following we represent our problems of $VCwnC$ in 2-SAT. We first remind the definition of 2-SAT problem. Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ be a finite set of boolean variables and $\mathcal{FORM} = C_1 \wedge \ldots \wedge C_l$ a formula composed of *clauses* $C_i = y_{i,1} \vee y_{i,2}$ for which $\forall (i,j)$, $y_{i,j}$ is a *literal* on a variable x_k i.e. whether x_k or $\neg x_k$. A *truth assignment* $\tau : \mathcal{X} \rightarrow \{0,1\}$ *satisfies* $\mathcal{FORM}$ if and only if each clause C_i is true, i.e. $\tau(y_{i,1}) = 1$ or $\tau(y_{i,2}) = 1$ (where, by natural extension on τ, if $y_{i,j} = \neg x_k$, $\tau(y_{i,j}) = \neg \tau(x_k)$). A *solution* of 2-SAT is a truth assignment τ that satisfies $\mathcal{FORM}$. A *minimal truth assignment* τ^* is an assignment satisfying $\mathcal{FORM}$ with a minimal number of variables assigned to 1 (minimal number of $\tau^*(x_i) = 1$). We note $|\tau|$ the number of boolean variables such that $\tau(x_i) = 1$.

Melven Robert Krom proposed in [10] a polynomial algorithm by transitive closure to solve the *decisional* problem of 2-SAT (i.e. knowing whether there exists an assignment satisfying the formula and constructing one if it is the case). Aspvall, Plass and Tarjan [1] improved the efficiency of the approach in 1979. In 1992, Dan Gusfield and Leonard Pitt [5] proved that finding a minimal assignment is $\mathcal{NP}$*-hard*. In the same paper they present a 2-approximation algorithm for this problem. Now we make the transformations between the problems. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be any graph and $\mathcal{C}$ be a *set of conflicts*. Each vertex is associated to a boolean variable x_i (to simplify x_i will denote the variable and the associated vertex). For each edge $x_i x_j$ of $\mathcal{G}$, we add the "positive" clause $(x_i \vee x_j)$ in $\mathcal{FORM}$. For each conflict $\{x_i, x_j\} \in \mathcal{C}$, we add the "negative" clause $(\neg x_i \vee \neg x_j)$ in $\mathcal{FORM}$.

By construction, $\mathcal{FORM}$ is a 2-SAT formula. Let τ be a truth assignment of $\mathcal{FORM}$ and $\mathcal{S}$ be the set of vertices x_i of $\mathcal{G}$ such that $\tau(x_i) = 1$ (note that $|\mathcal{S}| = |\tau|$). We prove now that $\mathcal{S}$ is a $VCwnC$ of $\mathcal{G}$ if and only if τ satisfies $\mathcal{FORM}$.

Indeed, suppose τ satisfies $\mathcal{FORM}$; then all the positive clauses $(x_i \vee x_j)$ are satisfied and consequently all edges of $\mathcal{E}$ are covered by $\mathcal{S}$. Similarly, negative clauses $(\neg x_i \vee \neg x_j)$ are satisfied and consequently for each conflict in $\mathcal{C}$ there is at most one of these two vertices in $\mathcal{S}$. Thus $\mathcal{S}$ is a $VCwnC$ of $\mathcal{G}$. Conversely if $\mathcal{S}$ is a $VCwnC$ of $\mathcal{G}$ then all edges of $\mathcal{E}$ have at least one vertex in $\mathcal{S}$ and then positive clauses $(x_i \vee x_j)$ are satisfied. Moreover, as each conflict of $\mathcal{C}$ has at most one vertex in $\mathcal{S}$ then negative clauses $(\neg x_i \vee \neg x_j)$ are satisfied. Hence truth assignment τ satisfies $\mathcal{FORM}$.

By the previous polynomial transformations and by using algorithms of [10] or [1], given $\mathcal{G}$ and $\mathcal{C}$ it is polynomial to determine whether $\mathcal{G}$ contains a $VCwnC$ and in the positive case to construct one.

It is easy to prove that $\mathcal{S}$ is an *optimal* $VCwnC$ of $\mathcal{G}$ if and only if τ is a minimal assignment, since by construction $|\mathcal{S}| = |\tau|$. Hence, finding an optimal $VCwnC$ is equivalent to construct a minimal truth assignment. Using our previously described polynomial transformation and the 2-approximation of [5] for the minimal assignment problem, we naturally get a 2-approximation for our minimal $VCwnC$ problem. Based on all these elements we get:

Theorem 3.1 *Given* $(\mathcal{G}, \mathcal{C})$ *it is polynomial to construct a* VCwnC *iff there exists one. There is a* 2*-approximation algorithm for the problem of minimizing its size.*

To finish we can mention that these results can be extended to the more general weighted cases.

3.2 Connected Vertex Cover With no conflict

A *Connected Vertex Cover with no conflict* ($CVCwnC$) in $(\mathcal{G}, \mathcal{C})$ is a vertex cover $\mathcal{S}$ of $\mathcal{G}$ such that $\mathcal{G}[\mathcal{S}]$ is connected and such that no pair of vertices of $\mathcal{S}$ is in $\mathcal{C}$. In this Section we prove that deciding whether there exists a $CVCwnC$ is $\mathcal{NP}$*-complete*. Note that when $\mathcal{C} = \emptyset$ (i.e. the "classical" version), there exists a connected vertex cover iff $\mathcal{G}$ is connected.

We make a reduction to the well-known $\mathcal{NP}$*-complete* problem *Exact 3-Cover* (*X3C*) [4]: Let $\mathcal{X} = \{u_1, u_2, \ldots, u_{3q}\}$ be a set of elements and $\mathcal{H}$ be a set of k subsets $C_i \subseteq \mathcal{X}$ and $|C_i| = 3$ such that $\cup_{i=1}^{k} C_i = \mathcal{X}$. The *X3C* problem is to decide whether there is an exact cover $\mathcal{S}_{\mathcal{H}} \subseteq \mathcal{H}$ of $\mathcal{X}$: $\forall C_i, C_j \in \mathcal{S}_{\mathcal{H}}, i \neq j, C_i \cap C_j = \emptyset$ and $\displaystyle\bigcup_{C_i \in \mathcal{S}_{\mathcal{H}}} C_i = \mathcal{X}$.

Theorem 3.2 *The problem of deciding whether there exists a* CVCwnC *is* $\mathcal{NP}$-complete*, even in bipartite graphs.*

`Preuve.` The problem is clearly in $\mathcal{NP}$. Let $(\mathcal{X}, \mathcal{H})$ be any instance of *X3C*. We construct an instance $(\mathcal{G}, \mathcal{C})$ as follows: Each $u_i \in \mathcal{X}$ is a vertex of $\mathcal{G}$ (also noted u_i). Each $C_i \in \mathcal{H}$ is a vertex of $\mathcal{G}$ (also noted C_i). A new vertex r is added. Vertex r is connected to each C_i. In $\mathcal{G}$ each vertex C_i is connected to the 3 vertices u_a, u_b, u_c iff in $\mathcal{H}$ the set $C_i = \{u_a, u_b, u_c\}$. Two different vertices C_i and C_j are in conflict ($\{C_i, C_j\} \in \mathcal{C}$) iff in $\mathcal{H}$, $C_i \cap C_j \neq \emptyset$. This construction is polynomial and the graph $\mathcal{G}$ is bipartite.

Let $(\mathcal{X}, \mathcal{H})$ be any instance of *X3C* and $(\mathcal{G}, \mathcal{C})$ be its associated instance. Suppose there exists a solution $\mathcal{S}_{\mathcal{H}}$ for the *X3C* instance $(\mathcal{X}, \mathcal{H})$. Let us consider $\mathcal{S} = \{r\} \cup \mathcal{S}_{\mathcal{H}} \cup \mathcal{X}$. $\mathcal{G}[\mathcal{S}]$ is connected (vertices C_i are connected via $r \in \mathcal{S}$ and each u_i is connected to one C_j because $\mathcal{S}_{\mathcal{H}}$ covers $\mathcal{X}$). Each edge of $\mathcal{E}$ is covered by $\mathcal{S}$ (since $\mathcal{S}$ contains r and all the $u_i \in \mathcal{X}$). Hence $\mathcal{S}$ is a connected vertex cover of $\mathcal{G}$. Moreover there is no conflict in $\mathcal{S}$ since the conflicts only occur between vertices of type C_i that have at least one common vertex. But as $\mathcal{S}_{\mathcal{H}}$ is a solution of *X3C*, $\forall C_i, C_j \in \mathcal{S}_{\mathcal{H}}, i \neq j, C_i \cap C_j = \emptyset$. $\mathcal{S}$ is a $CVCwnC$.

Suppose now that $(\mathcal{G}, \mathcal{C})$ contains a $CVCwnC$ $\mathcal{S}$. As $\mathcal{G}[\mathcal{S}]$ is connected, $\mathcal{S}$ necessarily contains vertices of type C_i. Let $\mathcal{S}_{\mathcal{H}}$ be this set of vertices. As $\mathcal{S}$ contains no conflict we have: $\forall C_i, C_j \in \mathcal{S}_{\mathcal{H}}, C_i \cap C_j = \emptyset$. We also have $\bigcup_{C_i \in \mathcal{S}_{\mathcal{H}}} C_i = \mathcal{X}$, otherwise $\exists x \in \mathcal{X}$ such that $x \notin \bigcup_{C_i \in \mathcal{S}_{\mathcal{H}}} C_i$. This means that we are in one of the following two cases: (1) $x \notin \mathcal{S}$ but as x has at least one incident edge, this edge is not covered and this is a contradiction with the fact that $\mathcal{S}$ is a vertex cover. (2) $x \in \mathcal{S}$; in this case its incident edges are covered by $\mathcal{S}$ but x is then an isolated vertex of $\mathcal{G}[\mathcal{S}]$ (no neighbor of x is in $\mathcal{S}$) but this is in contradiction with the fact that $\mathcal{G}[\mathcal{S}]$ is connected. $\square$

3.3 Independent Dominating Set With no conflict

Given $(\mathcal{G}, \mathcal{C})$, an *Independent Dominating set with no conflict* ($IDwnC$) is a set $\mathcal{S} \subseteq \mathcal{V}$ such that $\mathcal{S}$ is a dominating set of $\mathcal{G}$ ($\forall u \in \mathcal{V}$, $u \in \mathcal{S}$ or $N(u) \cap \mathcal{S} \neq \emptyset$), $\mathcal{S}$ is an independent set of $\mathcal{G}$ (for all $u \in \mathcal{S}$ and all $v \in \mathcal{S}$, $uv \notin \mathcal{E}$) and $\mathcal{S}$ contains no conflict ($\forall u, v \in \mathcal{S}, \{u, v\} \notin \mathcal{C}$). It is known that in any graph $\mathcal{G}$ there is always an independent dominating set but in $(\mathcal{G}, \mathcal{C})$ an $IDwnC$ does not always exist.

Theorem 3.3 *The problem of deciding whether there exists an $IDwnC$ is* $\mathcal{NP}$-complete*, even in bipartite graphs.*

`Preuve.` The problem is clearly in $\mathcal{NP}$. We make a reduction to *X3C* (see Section 3.2 for notations). Let $(\mathcal{X}, \mathcal{H})$ be any instance of *X3C*. We construct an instance $(\mathcal{G}, \mathcal{C})$ of our problem: Each $u_i \in \mathcal{X}$ is a vertex of $\mathcal{G}$ (also noted u_i). Each $C_i \in \mathcal{H}$ is a vertex of $\mathcal{G}$ (also noted C_i). Each vertex C_i is connected to a new vertex r_i (only connected to C_i). In $\mathcal{G}$ each vertex C_i is connected to the 3 vertices u_a, u_b, u_c iff in $\mathcal{H}$ the set $C_i = \{u_a, u_b, u_c\}$. Let us construct $\mathcal{C}$: Two different vertices C_i and C_j are in conflict ($\{C_i, C_j\} \in \mathcal{C}$) iff in $\mathcal{H}$, $C_i \cap C_j \neq \emptyset$. Each vertex u_i is in conflict with *all* the others vertices of $\mathcal{G}$. The construction of the instance $(\mathcal{G}, \mathcal{C})$ is polynomial and graph $\mathcal{G}$ is bipartite.

Figure 2 shows an example of construction of the graph $\mathcal{G}$ from a *X3C* instance: $\mathcal{X} = \{u_1, \ldots, u_9\}$ and $C_1 = \{u_1, u_2, u_3\}$, $C_2 = \{u_3, u_4, u_5\}$, $C_3 = \{u_5, u_6, u_7\}$ and $C_4 = \{u_4, u_8, u_9\}$.

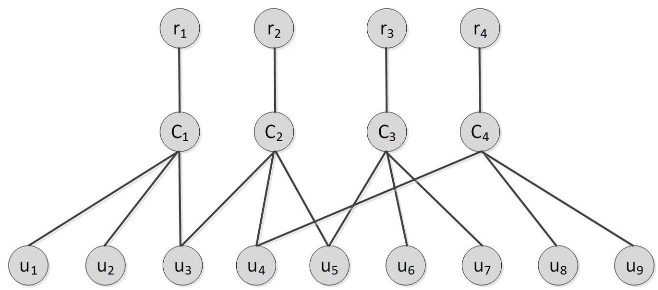

Figure 2: Graph $\mathcal{G}$ from an instance of *IDwFaRV*.

Suppose there exists a solution $\mathcal{H}$' for the *X3C* instance $(\mathcal{X}, \mathcal{H})$. Let $\mathcal{S}$ be the set composed of all the vertices $C_i \in \mathcal{H}'$ and for each $C_j \notin \mathcal{H}'$ we add to $\mathcal{S}$ its neighbor r_j. Clearly $\mathcal{S}$ is an independent set of $\mathcal{G}$. $\mathcal{S}$ is also a dominating set of $\mathcal{G}$ (each u_i is dominated by exactly one $C_j \in \mathcal{H}'$, each $C_j \notin \mathcal{H}'$ is dominated by $r_j \in \mathcal{S}$ and each r_i is either in $\mathcal{S}$ or dominated by its neighbor C_i). $\mathcal{S}$ contains no conflict (since $\mathcal{H}'$ is a partition of $\mathcal{X}$ and vertices of type C_i are in conflict together if and only if they are not disjoint).

Suppose now that there exists $\mathcal{S}$ an *IDwnC* of $(\mathcal{G}, \mathcal{C})$. As $\mathcal{S}$ contains no conflict, it contains no vertex of $\mathcal{X}$ (that are in conflict with all the other vertices). Hence it necessarily contains vertices of type C_i; Let $\mathcal{H}'$ be the set of these vertices. As $\mathcal{S}$ is a dominating set of $\mathcal{G}$, $\cup_{C_i \in \mathcal{H}'} C_i = \mathcal{X}$. Moreover, as $\mathcal{S}$ contains no conflict, for all $C_i \in \mathcal{H}'$ and all $C_j \in \mathcal{H}'$, $i \neq j$, we have $C_i \cap C_j = \emptyset$. Hence $\mathcal{H}$' is a solution of the *X3C* problem. $\square$

3.4 Dominating Set With no conflict

Given $(\mathcal{G}, \mathcal{C})$, a *Dominating set with no conflict* (*DwnC*) is a set $\mathcal{S} \subseteq \mathcal{V}$ such that $\mathcal{S}$ is a dominating set of $\mathcal{G}$ ($\forall u \in \mathcal{V}$, $u \in \mathcal{S}$ or $N(u) \cap \mathcal{S} \neq \emptyset$), and $\mathcal{S}$ contains no conflict ($\forall u, v \in \mathcal{S}, \{u, v\} \notin \mathcal{C}$). It is easy to see that in any graph $\mathcal{G}$ there is a dominating set but in $(\mathcal{G}, \mathcal{C})$ a *DwnC* does not always exist.

Theorem 3.4 *The problem of deciding whether there exists a DwnC is $\mathcal{NP}$-*complete*, even in bipartite graphs.*

`Preuve.` The problem is clearly in $\mathcal{NP}$. Let $(\mathcal{G}, \mathcal{C})$ be any instance of the problem *IDwnC*. Let $\mathcal{C}' = \mathcal{C} \cup \mathcal{E}$ (each edge uv of $\mathcal{G}$ is considered as a conflict $\{u, v\}$ in $\mathcal{C}'$). It is now easy to show that: $\mathcal{S}$ is an *IDwnC* of $(\mathcal{G}, \mathcal{C})$ if and only if $\mathcal{S}$ is a *DwnC* of $(\mathcal{G}, \mathcal{C}')$. This equivalence and Theorem 3.3 show the result. $\square$

3.5 Steiner Tree With no conflict

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be any graph, $\mathcal{C}$ be any set of conflicts of $\mathcal{G}$ and M be any subset of vertices of $\mathcal{G}$ ($M \subseteq \mathcal{V}$). A *Steiner Tree with no conflict* (*STwnC*) for the instance $(\mathcal{G}, \mathcal{C}, M)$ is a *tree* $T = (\mathcal{V}_T, \mathcal{E}_T)$ covering M ($M \subseteq \mathcal{V}_T \subseteq \mathcal{V}$) in $\mathcal{G}$ ($\mathcal{E}_T \subseteq \mathcal{E}$) containing no conflict of $\mathcal{C}$ (for all $u \in \mathcal{V}_T$ and all $v \in \mathcal{V}_T$, $\{u, v\} \notin \mathcal{C}$). Given only $\mathcal{G}$ and M it is polynomial to decide if there exists a tree covering M in $\mathcal{G}$. Adding conflicts leads to an $\mathcal{NP}$*-complete* problem.

Theorem 3.5 *The problem of deciding whether there exists a* STwnC *is* $\mathcal{NP}$-complete*, even in bipartite graphs.*

`Preuve.` The problem is in $\mathcal{NP}$. The reduction, again, is done with *X3C* (see Section 3.2 for notations). Let $(\mathcal{X}, \mathcal{H})$ be any instance of *X3C*. Let us construct an instance $(\mathcal{G}, \mathcal{C}, M)$ of *STwnC*: Each $u_i \in \mathcal{X}$ is a vertex of $\mathcal{G}$ (also noted u_i). Each $C_i \in \mathcal{H}$ is a vertex of $\mathcal{G}$ (also noted C_i). A new vertex r is added. Vertex r is connected to each C_i. In $\mathcal{G}$ each vertex C_i is connected to the 3 vertices u_a, u_b, u_c iff in $\mathcal{H}$ the set $C_i = \{u_a, u_b, u_c\}$. We set $M = \mathcal{X}$. Two different vertices C_i and C_j are in conflict ($\{C_i, C_j\} \in \mathcal{C}$) iff in $\mathcal{H}$, $C_i \cap C_j \neq \emptyset$. This construction is polynomial and the graph $\mathcal{G}$ is bipartite. Let $(\mathcal{X}, \mathcal{H})$ be any instance of *X3C* and $(\mathcal{G}, \mathcal{C}, M)$ the associated instance of *STwnC*.

To illustrate this construction, Figure 3 presents graph $\mathcal{G}$ from the following *X3C* instance: $X = \{u_1, \ldots, u_9\}$ and $C_1 = \{u_1, u_2, u_3\}$, $C_2 = \{u_3, u_4, u_5\}$, $C_3 = \{u_5, u_6, u_7\}$ and $C_4 = \{u_4, u_8, u_9\}$.

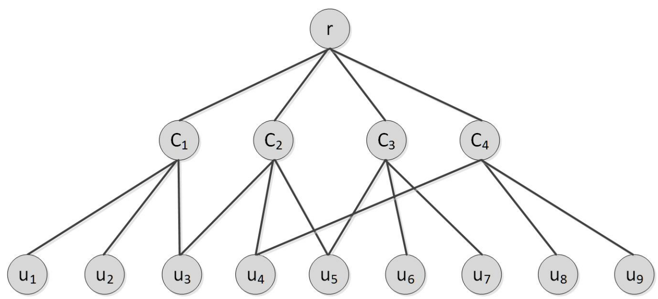

Figure 3: Graph $\mathcal{G}$ from an instance of *X3C*.

Suppose there exists a solution $\mathcal{S}_\mathcal{H}$ for the *X3C* problem. Let $\mathcal{S} = \{r\} \cup \mathcal{S}_\mathcal{H} \cup \mathcal{X}$. $\mathcal{S}$ contains all the vertices of M because $M = \mathcal{X} \subset \mathcal{S}$. $\mathcal{G}[\mathcal{S}]$ is connected: The C_i are connected via $r \in \mathcal{S}$ and each u_i is connected to a C_j because $\mathcal{S}_\mathcal{H}$ covers $\mathcal{X}$. Hence $\mathcal{G}[\mathcal{S}]$ contains a tree containing all the vertices of M. Moreover, there is no conflict since the only possible conflicts are between vertices of type C_i which are non disjoint and that is not the case here because $\mathcal{S}_\mathcal{H}$ is a partition of $\mathcal{X}$.

Suppose now that there exists $T = (\mathcal{V}_T, \mathcal{E}_T)$ a *STwnC* of $(\mathcal{G}, \mathcal{C}, M)$. As T is connected, it necessarily contains vertices of type C_i. Let $\mathcal{S}_\mathcal{H}$ be this set of vertices. As T contains no conflict we have: $\forall C_i, C_j \in \mathcal{S}_\mathcal{H}, i \neq j, C_i \cap C_j = \emptyset$. We also have $\bigcup_{C_i \in \mathcal{S}_\mathcal{H}} C_i = M = \mathcal{X}$. Otherwise $\exists x \in M$ such that $x \notin \bigcup_{C_i \in \mathcal{S}_\mathcal{H}} C_i$. But we are then in one of the following cases: (1) $x \notin T$, but this is in contradiction with the fact that T covers M; (2) $x \in T$; in this case, x is an isolated vertex of T but this is in contradiction with the fact that T is a tree. $\square$

4 Conclusion and perspectives

In Section 2 we proved that, given an instance $(\mathcal{G}, F, R)$, it is polynomial to decide whether it contains a vertex cover or a connected vertex cover or a dominating set or a Steiner tree containing all vertices of R and no vertices of F. Moreover, we proposed approximation algorithms to construct solutions of controlled size (compared to the optimal one) when it is possible. We also showed that it is $\mathcal{NP}$*-complete* to decide whether it contains an independent dominating set with all vertices of R and no vertices of F.

In Section 3 we proposed and solved a new variant for this 5 problems, namely constructing a solution containing at most one of two vertices that are in conflict (if one is in a solution, the other one cannot be). For the vertex cover with no conflict we shown that any instance can be transformed in an instance of a 2-SAT in polynomial time and that well-known algorithms for 2-SAT can be used. The optimization version can be treated by a 2-approximation algorithm. Unfortunately adding conflicts in other classical problems (dominating set, independent dominating set and Steiner tree) leads to problems where it is $\mathcal{NP}$*-complete* even to know whether there exists a solution.

Our models can be used to take into account practical constraints implying devices that are incompatible, regardless the nature of this compatibility. We can note that in case of conflicts the problems become much harder than their classical counterparts.

A first natural perspective is to do the same study to other optimization problems. A second perspective, that can be of theoretical interest, is to propose exact (but non polynomial) algorithms to solve the hard problems we obtained. In a more general algorithmic perspective, the same kind of study can be done by defining sets of forbidden and required *edges* instead of vertices for other problems of graphs.

References

[1] B. Aspvall, M. F. Plass, and R. E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Inf. Process. Lett.*, 8(3):121–123, 1979.

[2] A. Darmann, U. Pferschy, J. Schauer, and G. J. Woeginger. Paths, trees and matchings under disjunctive constraints. *Discrete Applied Mathematics*, 159(16):1726–1735, 2011.

[3] F. Delbot, C. Laforest, and R. Phan. New approximation algorithms for the vertex cover problem. In *IWOCA*, volume 8288 of *Lecture Notes in Computer Science*, pages 438–442. Springer, 2013.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman & Co Ltd, first edition edition, Jan. 1979.

[5] D. Gusfield and L. Pitt. A bounded approximation for the minimum cost 2-sat problem. *Algorithmica*, 8(2):103–117, 1992.

[6] D. S. Johnson. Approximation algorithms for combinatorial problems. In *Proceedings of the fifth annual ACM symposium on Theory of computing*, pages 38–49. ACM, 1973.

[7] M. M. Kanté, C. Laforest, and B. Momège. An exact algorithm to check the existence of (elementary) paths and a generalisation of the cut problem in graphs with forbidden transitions.

In *SOFSEM*, volume 7741 of *Lecture Notes in Computer Science*, pages 257–267. Springer, 2013.

[8] M. M. Kanté, C. Laforest, and B. Momège. Trees in graphs with conflict edges or forbidden transitions. In *TAMC (Theory and Applications of Models of Computation)*, volume 7876 of *Lecture Notes in Computer Science*, pages 343–354. Springer, 2013.

[9] J. Kováč. Complexity of the path avoiding forbidden pairs problem revisited. *Discrete Applied Mathematics*, 161(10–11):1506–1512, 7 2013.

[10] M. R. Krom. The decision problem for a class of first-order formulas in which all disjunctions are binary. *Mathematical Logic Quarterly*, 13(1-2):15–20, 1967.

[11] C. Laforest and B. Momège. Some hamiltonian properties of one-conflict graphs. In *Combinatorial Algorithms - 25th International Workshop, IWOCA 2014, Duluth, MN, USA, October 15-17, 2014, Revised Selected Papers*, pages 262–273, 2014.

[12] C. Laforest and B. Momège. Nash-williams-type and chvátal-type conditions in one-conflict graphs. In *SOFSEM 2015: Theory and Practice of Computer Science - 41st International Conference on Current Trends in Theory and Practice of Computer Science, Pec pod Sněžkou, Czech Republic, January 24-29, 2015. Proceedings*, pages 327–338, 2015.

[13] G. Robins and A. Zelikovsky. Improved steiner tree approximation in graphs. In *SODA*, pages 770–779. Citeseer, 2000.

[14] C. Savage. Depth-first search and the vertex cover problem. *Inf. Process. Lett.*, 14(5):233–235, July 1982.

[15] S. Szeider. Finding paths in graphs avoiding forbidden transitions. *Discrete Applied Mathematics*, 126(2-3):261–273, 2003.

[16] V. V. Vazirani. *Approximation algorithms.* Springer, 2001.