Finding d-Cuts in Probe H-Free Graphs

Konrad K. Dabrowski[1][0000−0001−9515−6945], Tala Eagling-Vose[2][0009−0008−0346−7032], Matthew Johnson[2][0000−0002−7295−2663], Giacomo Paesani[3][0000−0002−2383−1339], and Daniël Paulusma[2][0000−0001−5945−9287]

[1] School of Computing, Newcastle University, Newcastle, UK
konrad.dabrowski@newcastle.ac.uk
[2] Department of Computer Science, Durham University, Durham, UK
{tala.j.eagling-vose,matthew.johnson2,daniel.paulusma}@durham.ac.uk
[3] Department of Computer Science, Sapienza University of Rome, Rome, Italy
paesani@di.uniroma1.it

Abstract. For an integer $d \geq 1$, the d-Cut problem is that of deciding whether a graph has an edge cut in which each vertex is adjacent to at most d vertices on the opposite side of the cut. The 1-Cut problem is the well-known Matching Cut problem. The d-Cut problem has been extensively studied for H-free graphs. We extend these results to the probe graph model, where we do not know all the edges of the input graph. For a graph H, a partitioned probe H-free graph (G, P, N) consists of a graph $G = (V, E)$, together with a set $P \subseteq V$ of probes and an independent set $N = V \setminus P$ of non-probes such that we can change G into an H-free graph by adding zero or more edges between vertices in N. For every graph H and every integer $d \geq 1$, we completely determine the complexity of d-Cut on partitioned probe H-free graphs.

Keywords: d-cut · probe graph · subset problem · H-free graph

1 Introduction

When studying computationally hard problems for special graph classes, it is natural to generalize polynomial-time results for certain graph classes to larger graph classes. In particular, consider a graph H' and an induced subgraph H of H'. The class of H-*free* graphs (class of graphs that do not contain H as an induced subgraph) is contained in the class of H'-free graphs. Say an NP-complete problem Π is polynomial-time solvable on H-free graphs. Is Π also polynomial-time solvable on H'-free graphs? This question leads to complexity studies for a wide range of graph problems where the goal is to obtain *complexity dichotomies* that tell us for exactly which graphs H a certain NP-complete problem is polynomial-time solvable, and for which graphs H it stays NP-complete.

We follow this line of research, but also assume that we do not know all the edges of the input graph. Before explaining the latter in more detail, we first introduce the problem that we study. Consider a connected graph $G = (V, E)$.

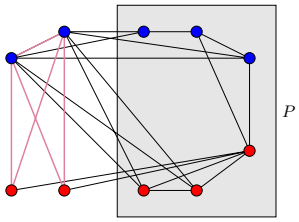

Fig. 1: A probe P_5-free graph G with a set P of probes. The edge of G are shown in black, whereas the edges in F (which do not belong to G) are shown in purple. Note that $G + F$ is indeed P_5-free and that the red-blue colouring corresponds to a 2-cut of G.

A subset $M \subseteq E$ is an *edge cut* of G if it is possible to partition V into two non-empty sets B (*blue* vertices) and R (*red* vertices) in such a way that M is the set of all edges with one end-vertex in B and the other in R. Now, for an integer $d \geq 1$, if every blue vertex has at most d red neighbours, and every red vertex has at most d blue neighbours, then the edge cut M is said to be a *d-cut* of G. See also Figure 1. The d-Cut problem is that of deciding whether a connected graph has a d-cut. A 1-cut is also called a *matching cut*, and the 1-Cut problem is better known as Matching Cut. For all $d \geq 1$, d-Cut is NP-complete [8, 14]. Graphs with matching cuts were introduced in 1970 by Graham [15] in the context of number theory; for other applications see [2, 9, 11, 25].

Our Focus. We consider the classical probe graph model, which was introduced by Zhang et al. [27] in 1994 to deal with partial information in genome research. In this model, the complete set of neighbours is only known for *some* vertices of the input graph G. These vertices form the set P of *probes*. The other vertices of G form the set N of *non-probes*. As we do not know the adjacencies between vertices in N, the set N is an independent set in G. However, in the probe graph model we also assume there exists a "certifying" set F of edges between (some of) the non-probes such that $G + F$ has some known global structure; again see Figure 1. In our paper, $G + F$ is H-free. Note that $G[P]$ is already H-free.

So, a *partitioned probe H-free* graph (G, P, N) consists of a graph $G = (V, E)$, a set $P \subseteq V$ of probes and an independent set $N = V \setminus P$ of non-probes, such that $G + F$ is H-free for some edge subset $F \subseteq \binom{N}{2}$. Any H-free graph is also (partitioned) probe H-free: take $P = V$ and $N = \emptyset$. Hence, (partitioned) probe H-free graphs contain all H-free graphs, and any NP-completeness results for H-free graphs carry over to partitioned probe H-free graphs. We therefore ask:

For which H, does d-Cut stay polynomial-time solvable on probe H-free graphs?

As such, our paper belongs to a recent systematic study of graph problems on probe H-free graphs. This study was initiated by Brettell et al. [5] for Vertex Cover, whereas the previous literature on probe graphs aimed to characterize and recognize classes of probe graphs; see e.g. [3, 6, 7, 12, 13]. For example, if $H = P_4$, then probe H-free graphs can be recognized in polynomial time [7].

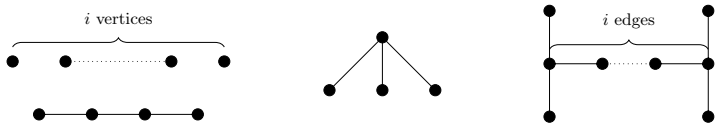

Fig. 2: The graphs $sP_1 + P_4$, $K_{1,3}$ and H_i^*, from left to right.

However, for most other graphs H, the complexity of recognizing probe H-free graphs is still unknown. Hence, for our algorithms, we assume that P and N are part of the input, that is, we will consider partitioned probe H-free graphs.

We will also consider two related problems: Maximum Matching Cut and Perfect Matching Cut on probe H-free graphs. The first is to decide if a connected graph has a matching cut of at least k edges for some integer k. The second is to decide if a connected graph has a *perfect matching cut*, that is, an edge cut that is a perfect matching. This problem is also NP-complete [16].

Known Results. For two vertex-disjoint graphs G_1 and G_2, let $G_1 + G_2 = (V(G_1) \cup V(G_2), E(G_1) \cup E(G_2))$. We let sG be the disjoint union of s copies of G. We write $G_1 \subseteq_i G_2$ if G_1 is an induced subgraph of G_2. Let C_s denote the cycle on s vertices, P_t the path on t vertices, and $K_{1,r}$ the star on $r+1$ vertices. The graph $K_{1,3}$ is known as the *claw*. Let H_1^* be the "H"-graph, which has vertices u, v, w_1, w_2, x_1, x_2 and edges $uv, uw_1, uw_2, vx_1, vx_2$. For $i \geq 2$, let H_i^* be obtained from H_1^* by subdividing uv exactly $i-1$ times. See Figure 2.

In Theorems 1–3 we present the state-of-art for d-Cut, Perfect Matching Cut and Maximum Matching Cut for H-free graphs. Only Theorem 3 is a full dichotomy. The references in Theorem 1 are explained in [20] except for the recent result that 2-Cut is NP-complete for claw-free graphs [1]; note the jump in complexity from $d = 1$ to $d = 2$ for $H = 3P_2$ and $H = K_{1,3}$. For $d \geq 2$, the only three non-equivalent open cases in Theorem 1 are $H = 2P_4$, $H = P_6$ and $H = P_7$ (see also [20]). The references in Theorem 2 are explained in [23].

Theorem 1 ([1, 4, 8, 10, 17, 19–22, 24]). *Let H be a graph and $d \geq 1$.*

- *If $d = 1$, then d-Cut on H-free graphs is polynomial-time solvable if $H \subseteq_i sP_3 + S_{1,1,2}$, $sP_3 + P_4 + P_6$, or $sP_3 + P_7$ for some $s \geq 0$; and NP-complete if $H \supseteq_i K_{1,4}$, P_{14}, $2P_7$, $3P_5$, C_r for $r \geq 3$, or H_i^* for $i \geq 1$.*
- *If $d \geq 2$, then d-Cut on H-free graphs is polynomial-time solvable if $H \subseteq_i sP_1 + P_3 + P_4$ or $sP_1 + P_5$ for some $s \geq 0$; and NP-complete if $H \supseteq_i K_{1,3}$, $3P_2$, C_r for $r \geq 3$, or H_i^* for $i \geq 1$.*

Theorem 2 ([10, 17, 18, 22]). Perfect Matching Cut *on H-free graphs is polynomial-time solvable if $H \subseteq_i sP_4 + S_{1,2,2}$ or $sP_4 + P_6$ for some $s \geq 0$; and NP-complete if $H \supseteq_i K_{1,4}$, P_{14}, $2P_7$, $3P_6$, C_r for $r \geq 3$ or H_j^* for $j \geq 1$.*

Theorem 3 ([23]). Maximum Matching Cut *on H-free graphs is polynomial-time solvable if $H \subseteq_i sP_2 + P_6$ for some $s \geq 0$; and NP-complete otherwise.*

Our Results. We combine NP-completeness results from Theorems 1–3 with new polynomial and hardness results (shown in Section 3 and 4, resp.) to prove:

Theorem 4. *For a graph H, the following four complete dichotomies hold:*

- 1-Cut, Perfect Matching Cut *and* Maximum Matching Cut *on partitioned probe H-free graphs are polynomial-time solvable if $H \subseteq_i sP_1 + P_4$ for some $s \geq 0$; and* NP*-complete otherwise;*
- *for $d \geq 2$, d-*Cut *on partitioned probe H-free graphs is polynomial-time solvable if $H \subseteq_i P_1 + P_4$; and* NP*-complete otherwise.*

From Theorems 1 and 4, it follows that d-Cut becomes harder for *probe* H-free graphs than for H-free graphs (if P $\neq$ NP) even if $H = 2P_2$ for $d \geq 1$ and $H = 4P_1$ for $d \geq 2$.

2 Preliminaries and Basic Results

Throughout the paper, we only consider finite, undirected graphs without multiple edges and self-loops. We first define some general graph terminology.

Let $G = (V, E)$ be a graph. We let $N_G(v) = \{u \in V \mid uv \in E\}$ be the *(open) neighbourhood* of v and $N_G[v] = N_G(v) \cup \{v\}$ be the *closed neighbourhood* of v. Let $S \subseteq V$. We write $G[S]$ to denote the subgraph of G induced by S. A vertex $v \notin S$ is *complete* to S if $N(v) \supseteq S$, and v is *anti-complete* to S if $N(v) \cap S = \emptyset$. Let $S' \subseteq V$ with $S' \cap S = \emptyset$. If every vertex of S is complete (anti-complete) to S', then S is *complete* (*anti-complete*) to S'.

In our paper we also define some other probe graph classes. For example, we may say that a graph G with a set P of probes and a set N of non-probes is *probe split* if there exists a set $F \subseteq \binom{N}{2}$ such that $G + F$ is a split graph (a graph whose vertex set can be partitioned into a clique and an independent set).

We now recall some colouring terminology for d-cuts from [20] that is commonly used in the context of matching cuts (see, e.g. [21]). A *red-blue colouring* of a graph G colours every vertex of G either red or blue. For $d \geq 1$, a red-blue colouring is a *red-blue d-colouring* if every blue vertex has at most d red neighbours, every red vertex has at most d blue neighbours, and G has at least one blue vertex and at least one red vertex. See Figure 1 for red-blue d-colourings for $d = 2$ and $d = 3$. For some $d \geq 1$, a red-blue d-colouring is *perfect* if and only if every red vertex has exactly d blue neighbours and vice versa. This gives us the following straightforward observation (in the case of perfectness we focus on $d = 1$: a perfect 1-cut is a perfect matching cut).

Observation 5 ([20]) *For every $d \geq 1$, a connected graph G has a (perfect) d-cut if and only if G has a (perfect) red-blue d-colouring.*

Let $d \geq 1$. Let $G = (V, E)$ be a connected graph and $X, Y \subseteq V$ be disjoint sets. A *red-blue (X, Y)-d-colouring* of G is a red-blue d-colouring of G that colours all the vertices of X red and all the vertices of Y blue. We say that

(X, Y) is a *d-precoloured pair* of G. We will usually "guess" such a pair (X, Y) as the starting point in our algorithms. On a d-precoloured pair (X, Y) we can safely apply the following two rules exhaustively.

R1. Return `no` (i.e. G has no red-blue (X, Y)-d-colouring) if a vertex $v \in V$ is adjacent to $d+1$ vertices in X as well as to $d+1$ vertices in Y.
R1. Let $v \in V \setminus (X \cup Y)$. If v is adjacent to $d+1$ vertices in X, then put v in X, and if v is adjacent to $d+1$ vertices in Y, then put v in Y.

Afterwards, we either returned `no`, or we obtained two new sets $X' \supseteq X$ and $Y' \supseteq Y$. In the latter case we say that we have *colour-processed* (X, Y) *into* (X', Y'). By construction, every vertex of $V \setminus (X' \cup Y')$ is adjacent to at most d vertices of X' and to at most d vertices of Y'. The next lemma shows that we can work safely with (X', Y') instead of (X, Y).

Lemma 6 ([20]). *Let G be a connected graph with a precoloured pair (X, Y). It is possible, in polynomial time, to either colour-process (X, Y) into a pair (X', Y') such that G has a red-blue (X, Y)-d-colouring if and only if it has a red-blue (X', Y')-d-colouring, or to find that G has no red-blue (X, Y)-d-colouring.*

3 Polynomial-Time Results

In this section, we show our polynomial-time results for Maximum Matching Cut, Perfect Matching Cut and d-Cut ($d \geq 1$). That is, we show that Maximum Matching Cut (and thus 1-Cut) and Perfect Matching Cut are polynomial-time solvable on $(sP_1 + P_4)$-free graphs, and d-Cut, for $d \geq 2$, is polynomial-time solvable on $(P_1 + P_4)$-free graphs.

Our proofs are based on combining colour-processing with the observation that we can guess the closed neighbourhood of any set of size at most some constant c: this will lead to only $\mathcal{O}(2^c n^{cd})$ branches, due to the fact that any vertex can have at most d neighbours of the opposite colour. In our algorithm we choose a constant number of constant-sized sets in such a way that afterwards the whole input graph is coloured. We show below how we do the above for our most involved result, which is our algorithm for d-Cut for $(P_1 + P_4)$-free graphs for $d \geq 2$, and we omit the proof of Theorem 7 due to page restrictions.

Theorem 7. Maximum Matching Cut *and* Perfect Matching Cut *are polynomial-time solvable on partitioned probe (sP_1+P_4)-free graphs for all $s \geq 0$.*

To prove our next result (Theorem 9), we need a lemma whose proof we omit.

Lemma 8. *For any red-blue d-colouring of a connected P_4-free graph, some colour class has size at most $2d$.*

Theorem 9. *For every $d \geq 2$, d-Cut is polynomial-time solvable on partitioned probe $(P_1 + P_4)$-free graphs.*

Proof. Let $d \geq 2$. Let $G = (V, E)$. Let (G, P, N) be a connected partitioned probe $(P_1 + P_4)$-free graph, so there exists an edge subset $F \subseteq \binom{N}{2}$ such that $G + F$ is $(P_1 + P_4)$-free. We show how to find a d-cut of G or that none exists.

Suppose that there is a set $Q \subseteq P$ that induces a P_4 in G. Then Q dominates G, otherwise G contains an induced $P_1 + P_4$ of which at most one vertex belongs to N and so $G + F$ also contains an induced $P_1 + P_4$. For every red-blue d-colouring of G, every vertex $v \in V$ has at most d neighbours in a different colour class. That is, there are $\mathcal{O}(n^d)$ red-blue d-colourings of $N_G[v]$. As $|Q| = 4$, we can consider all $\mathcal{O}(n^d)$ colourings of $N_G[Q]$, and, as Q dominates G, this is all red-blue d-colourings of G; hence, we can solve the problem in polynomial time.

We may now assume that $G[P]$ is a P_4-free graph (cograph) and we consider three cases according to the number of connected components of $G[P]$. As G is connected and N is an independent set of G, we find that P dominates N, that is, every vertex of N has a neighbour in P. This means that G has a red-blue d-colouring that colours every vertex of P blue only if G has a red-blue d-colouring that colours exactly one vertex of N red and all other vertices of G blue. We can check this in polynomial time. Hence, from now on, we will assume that in every red-blue d-colouring of G (if such a colouring exists), there is at least one red vertex and at least one blue vertex in P.

Case 9.1: $G[P]$ has exactly one connected component.
By Lemma 8, we may assume that a set X of at most $2d$ vertices of P is coloured red. We guess X. As we may assume that P is not monochromatic, we may assume that X is a proper non-empty subset of P. We colour every vertex of $P \setminus X$ blue. Then we consider all $\mathcal{O}(n^{2d})$ possible red-blue colourings of the neighbourhood of X in N. For each of them, we consider the remaining uncoloured vertices in N. As these only have blue neighbours, we can safely colour them blue. It remains to check in polynomial time if the obtained colouring of G is indeed a red-blue d-colouring.

Case 9.2: $G[P]$ has exactly two connected components.
The proof of this case has been omitted.

Case 9.3: $G[P]$ has at least three connected components.
Let the connected components of $G[P]$ have vertex sets $C_1, \ldots, C_r$ for some $r \geq 3$. We partition N into four types of vertices. Let $v \in N$. If v is complete to P, then we say that v is of *type-A*.

Now, suppose that v is not of type-A, but that v has a neighbour in every component of $G[P]$. We say that v is of *type-B* and make the following observation. We know that v is not complete to some set C_i, so we can find adjacent vertices x_i and x_i' in C_i such that x_i but not x_i' is adjacent to v. Suppose that v is not complete to another set C_j, so there is a vertex x_j' in C_j not adjacent to v. Let x_k be a vertex adjacent to v in the vertex set of a third component. We now have that $\{x_j'\} \cup \{x_i', x_i, v, x_k\}$ forms an induced $P_1 + P_4$ in G, and consequently in $G + F$, as it only contains one vertex of N, namely v. Hence, a type-B vertex is complete to all but one set C_i, in which it has least one neighbour.

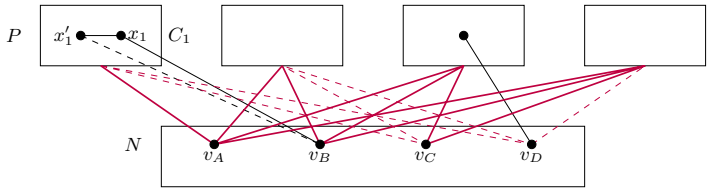

Fig. 3: Illustration of the types described in Case 9.3, specified by vertex label. A (dashed) purple line between a vertex $v \in N$ and some C_i represents that v is (anti-)complete to C_i. A black (dashed) edge between a vertex $v \in N$ and a vertex in P represents a (non)-edge.

Now, suppose that v is anti-complete to some C_h, so v is neither of type-A nor of type-B. We say that v is of *type-C* if v has a neighbour in at least two other components C_i and C_j. Suppose that v has a non-neighbour in either C_i or C_j, say in C_i. Let $x_h \in C_h$, $x_i, x_i' \in C_i$ such that $x_i x_i'$ is an edge with only x_i adjacent to v, and let $x_j \in C_j$ be adjacent to v. Now, $\{x_h\} \cup \{x_i', x_i, v, x_j\}$ induces a $P_1 + P_4$ in G, and also in $G + F$ as it has only one vertex from N, a contradiction as $G + F$ is $(P_1 + P_4)$-free. Hence, v is complete to every C_j in which it has a neighbour. So, a type-C vertex of N is complete to at least two and at most $r - 1$ sets in $\{C_1, \ldots, C_r\}$ and anti-complete to all other sets in $\{C_1, \ldots, C_r\}$.

Finally, if v is neither type-A nor type-B nor type-C, then v is of type-D. As G is connected, a type-D vertex v has at least one neighbour in one set C_i and is anti-complete to every other set in $\{C_1, \ldots, C_r\}$. See Figure 3 for an illustration. We distinguish between the following three cases:

Case 9.3.1 N has a type-A vertex.
Let $v \in N$ be of type-A. We colour v blue and we consider all $\mathcal{O}(n^d)$ possible red-blue colourings of its neighbourhood $N(v) = P$. We then find a set $Q \subseteq P$ of at most d vertices in P that are coloured red. As we already checked whether G has a red-blue d-colouring in which P is monochromatic, we may assume that Q is a proper non-empty subset of P. We now consider all possible $\mathcal{O}(n^{d^2})$ possible red-blue colourings of the neighbourhood of Q in N. For each one of them, the uncoloured vertices in N only have blue neighbours and form an independent set, we can safely colour them blue. Note that at least one vertex is red and at least one is blue. It remains to check whether the obtained colouring of G is a red-blue d-colouring. If so, we are done, and otherwise we discard the branch. Hence, as the number of branches is polynomial, and we can process each branch in polynomial time, this case takes polynomial time.

Case 9.3.2 N has no type-A vertices, but N has a type-B vertex.
Let $v \in N$ be of type-B. We assume without loss of generality that v has a neighbour and a non-neighbour in C_1 and is complete to $C_2, \ldots, C_r$. We colour v blue, and we consider all $\mathcal{O}(n^d)$ options of choosing a set X_v of at most d red neighbours of v in P. We colour all other neighbours of v in P blue. Hence,

afterwards part of C_1 is coloured, while all vertices of every C_i with $i \in \{2, \ldots, r\}$ are coloured. Moreover, by Lemma 8, we find that any red-blue d-colouring of G (if one exists) either colours at most $2d$ vertices of C_1 red, or else it colours at most $2d$ vertices of C_1 blue. Hence, we also consider all $\mathcal{O}(n^{2d})$ options for choosing a set $X \subseteq C_1$ of size at most $2d$ that consists of these vertices. We colour the vertices of $C_1 \setminus X$ with the opposite colour. We also colour the neighbourhood of $X_v \cup X$ in N in every possible way. This yields another $\mathcal{O}(n^{3d^2})$ branches. For each branch, we colour-process. If afterwards we find an uncoloured vertex in N whose neighbourhood in P is monochromatic, then we give it the unique colour of the vertices in its neighbourhood in P.

Suppose that afterwards there still exist uncoloured vertices in N. Let b be an arbitrary uncoloured vertex in N and note that b is adjacent to both a blue vertex x and a red vertex y in P. As we coloured the neighbourhood of $X_v \cup X$, neither x nor y belongs to $X_v \cup X$. As X_v contains all red vertices in $N(v)$, it follows that y belongs to $C_1 \setminus N(v)$, and thus $y \in C_1 \setminus X$. Consequently, all vertices of X are blue. As x, which is blue, is not in X, and all vertices in $C_1 \setminus X$ are red, x belongs to some set C_i with $i \geq 2$. Hence, all uncoloured vertices have a neighbour in C_1 and a neighbour in at least one other C_i. As N has no type-A vertices, every uncoloured vertex in N is of type-B or of type-C.

First, suppose that N has an uncoloured vertex b that is of type-B. By definition, b is complete to $r-1$ sets in $\{C_1, \ldots, C_r\}$. Let Y be the union of these $r-1$ sets. We recall that b is uncoloured and that all vertices in every C_i are coloured. Hence, Y must contain at most $2d$ vertices in total, otherwise we would have given b a colour during colour-processing. We now consider all $\mathcal{O}(n^{2d^2})$ possible red-blue colourings of the neighbourhood of Y in N. Afterwards, there are no uncoloured vertices left, as every uncoloured vertex was of type-B or of type-C and thus must have a neighbour in Y. It remains to check whether the obtained colouring of G is a red-blue d-colouring. If so, we are done, and otherwise we discard the branch.

So we can now assume that all uncoloured vertices are of type-C. Let b again denote an uncoloured vertex in N. From the definition of type-C, it follows that b is either complete or anti-complete to every set C_i. Recall that all uncoloured vertices of N, and thus b, have a neighbour in C_1. Hence, b is complete to C_1. Note that b is anti-complete to X, as otherwise b would have been coloured. This means that $X = \emptyset$, and thus every vertex of C_1 is coloured red. As b is uncoloured and we have colour-processed, this means that C_1 has size at most d. Hence, we can consider all $\mathcal{O}(n^{d^2})$ possible red-blue colourings of the neighbourhood of C_1 in N. Afterwards, all uncoloured vertices in N have received a colour (as they were all complete to C_1). It remains to check whether the obtained colouring of G is a red-blue d-colouring. If so, we are done, otherwise we discard the branch.

As the number of branches is polynomial, and we can process each branch in polynomial time, our algorithm takes polynomial time if this case occurs.

Case 9.3.3 N has no type-A and no type-B vertices.
As G is connected and $r \geq 3$, G must have vertices that are adjacent to at least

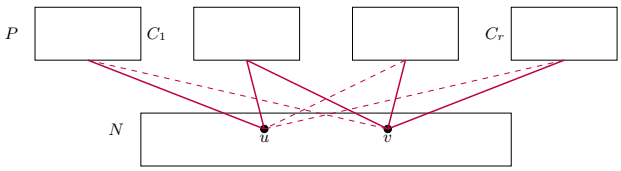

Fig. 4: A P-dominating pair $\{u, v\}$.

two sets C_i. As G has no vertices of type-A or type-B, all of these vertices must be of type-C.

We now make a useful observation. Let u and v be two type-C vertices, such that the following holds:

(i) u is complete to some C_h, but anti-complete to C_i;
(ii) v is anti-complete to C_h, but complete to C_i; and
(iii) u and v are both complete to some C_j.

In this case, we claim that $\{u, v\}$ is a *P-dominating pair*, that is, every C_i is complete to a least one of u, v; see Figure 4. For a contradiction, suppose that neither u nor v is complete to some C_k (so both are anti-complete to C_k as they are of type-C). Let $z \in C_k$. If $uv \notin E(F)$, then a vertex of C_h, u, a vertex of C_j and v form, together with z, an induced $P_1 + P_4$ in $G + F$. If $uv \in E(F)$, then a vertex of C_h, u, v and a vertex of C_i, together with z, form an induced $P_1 + P_4$ in $G + F$. So, in both cases, we derive a contradiction.

We now continue with the description of our algorithm. We choose a type-C vertex $v \in N$ such that v is complete to a maximum number of sets C_i over all type-C vertices of N. We say that v is of *maximum* type-C. From the definition of type-C, it follows that v is anti-complete to at least one set in $\{C_1, \ldots, C_r\}$. Let C_h be such a set. As G is connected, G contains a path from v to the vertices in C_h. Hence, without loss of generality, there exists a type-C vertex $u \in N$ that is complete to C_h and to at least one other C_j to which v is also complete. As v is of maximum type-C and v is not complete to C_h, to which u is complete, we find that u is also anti-complete to some set C_i to which v is complete. Hence, $\{u, v\}$ satisfies (i)–(iii), so $\{u, v\}$ is a P-dominating pair.

As $\{u, v\}$ is a P-dominating pair, we can colour P as follows. First suppose u and v are coloured alike, say both are coloured blue. We proceed in exactly the same way as in Case 9.1. As u and v can each have at most d red neighbours, we may assume that a set X of at most $2d$ vertices of P is coloured red. We guess X. As we already checked whether G has a red-blue d-colouring in which P is monochromatic, we may assume that X is a proper non-empty subset of P. We colour every vertex of $P \setminus X$ red. Afterwards, we consider all $\mathcal{O}(n^{2d})$ possible red-blue colourings of the neighbourhood of X in N. For each one of them, we consider the uncoloured vertices in N. As these only have blue neighbours, we can safely colour them blue. Note that at least one vertex is red and at least one vertex is blue. It remains to check in polynomial time whether the obtained colouring of G is indeed a red-blue d-colouring.

Now suppose u is coloured red and v is coloured blue. We guess a set X_u of at most d blue neighbours of u in P. We colour all other neighbours of u in P red. Similarly, we guess a set X_v of at most d blue neighbours of v in P. We colour all other neighbours of v in P red. This gives us $\mathcal{O}(n^{2d})$ branches. We note that every vertex of P has been coloured, as $\{u, v\}$ is a P-dominating pair. We now colour the neighbourhood of $X_u \cup X_v$ in N in every possible way. This gives us a further $\mathcal{O}(n^{2d^2})$ branches.

We let $C_1^*, \ldots, C_q^*$ be those sets C_i that contain a vertex of $X_u \cup X_v$. We let $C_1^u, \ldots, C_s^u$ be those sets C_i that contain no vertex of $X_u \cup X_v$ and are coloured red. We let $C_1^v, \ldots, C_t^v$ be those sets C_i that contain no vertex of $X_u \cup X_v$ and are coloured blue. Note that every C_i belongs to one of these three families of sets, but some of these families might be empty.

As v is coloured blue, its neighbours not in X_v are blue. Hence, v is anti-complete to every C_i^u. Similarly, u is anti-complete to every C_j^v. Recall that $\{u, v\}$ is a P-dominating pair and that each of u, v, being of type-C, is either complete or anti-complete to a set C_i. Consequently, u is complete to every C_i^u and v is complete to every C_i^v.

If C_1^u exists, then we select an arbitrary vertex $x_1 \in C_1^u$, and we colour the neighbourhood of x_1 in N in every possible way. This leads to $\mathcal{O}(n^d)$ additional branches. We do the same if C_1^v exists, leading to another $\mathcal{O}(n^d)$ branches. We now colour-process. Afterwards, we colour any vertex in N with monochromatic neighbourhood in P with the unique colour of its neighbours in P.

We claim that every vertex of type-D has now been coloured. For a contradiction, suppose $z \in N$ is of type-D and has no colour yet. We recall that all C_i^u and C_j^v are monochromatic and that z, being type-D, only has neighbours in exactly one C_i. This means that z must have both a blue neighbour and a red neighbour in some C_i^*. However, one of these two neighbours of z belongs to $X_u \cup X_v$, meaning z would have been coloured.

Hence, the only vertices of G that are possibly still uncoloured are type-C vertices in N. Let $b \in N$ be an uncoloured vertex of type-C. This means that b has both a red neighbour and a blue neighbour in P. By definition, b is either complete or anti-complete to C_i for every $i \in \{1, \ldots, r\}$. This means that b is anti-complete to every C_h^*, as we coloured the neighbourhood of $C_h^* \cap (X_u \cup X_v)$, which is a non-empty set by definition. Hence, the red neighbour of b must be some $z^u \in C_i^u$, and the blue neighbour of b must be some $z^v \in C_j^v$. From the above, we find that b is anti-complete to C_1^u and C_1^v (as we coloured the neighbourhood of one of the vertices in them). Hence, we have $i \geq 2$ and $j \geq 2$. Let $y^u \in C_1^u$ and $y^v \in C_1^v$. If $bv \notin F$, then $\{y^u\} \cup \{z^u, b, z^v, v\}$ induces a $P_1 + P_4$ in $G + F$. If $bv \in F$, then $\{y^u\} \cup \{z^u, b, v, y^v\}$ induces a $P_1 + P_4$ in $G + F$. See Figure 5 for an illustration. In both cases, we obtain a contradiction.

From the above, we conclude that there are no uncoloured vertices, and we have obtained a red-blue colouring of G. It now remains to check in polynomial time whether this is a red-blue d-colouring. If so, then we are done. Otherwise, we discard this branch and move on to the next branch. As the number of branches

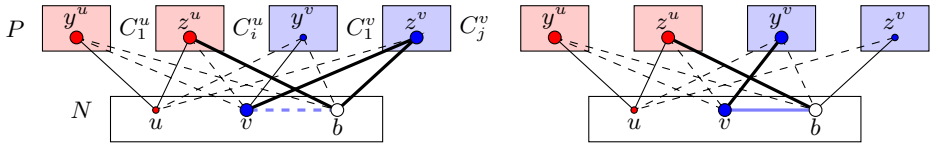

Fig. 5: Illustration of Case 9.3, where there is a P-dominating pair $\{u, v\}$ made of vertices of type-C and another type-C vertex b that has not been coloured yet. The graph $G + F$ contains an induced $P_1 + P_4$ (highlighted by large vertices and thick edges) regardless of whether there is an edge between v and b or not.

is polynomial, and we can process each branch in polynomial time, our algorithm runs in polynomial time (its correctness follows from its description). □

4 NP-Completeness Results

To finish the proof of Theorem 4, we must show that 1-Cut and Perfect Matching Cut are NP-complete on probe $2P_2$-free graphs and probe $K_{1,3}$-free graphs and that for $d \geq 2$, d-Cut is NP-complete on probe $2P_2$-free graphs and probe $4P_1$-free graphs. All other NP-completeness results follow directly from the corresponding NP-completeness results in Theorems 2 and 3.

We start with the following result, which we recall is in contrast to the polynomial-time result of 1-Cut for $K_{1,3}$-free graphs due to Bonsma [4]. We omit its proof which is based on an observation of Moshi [24].

Theorem 10. 1-Cut *is* NP*-complete on probe* $K_{1,3}$*-free graphs.*

The diamond $\overline{2P_1 + P_2}$ is obtained from taking the complement of $2P_1 + P_2$, or equivalently, from the K_4 after removing an edge. A graph is *subcubic* if it has maximum degree at most 3. We omit the proof of our next result as well, as it is similar to a corresponding result for Vertex Cover from [5] except that we reduce from a different known NP-complete problem.

Theorem 11. Perfect Matching Cut *is* NP*-complete on the class of probe* $(K_{1,3}, \overline{2P_1 + P_2})$*-free subcubic planar graphs (and thus on probe* $K_{1,3}$*-free graphs).*

We recall that a graph is split if and only if it is $(2P_2, C_4, C_5)$-free, and we show:

Theorem 12. *For every* $d \geq 1$*,* d-Cut *and* Perfect Matching Cut *are* NP*-complete on probe split graphs (and thus on probe* $2P_2$*-free graphs).*

Proof. It is known that for every $d \geq 1$, d-Cut and Perfect Matching Cut are NP-complete for bipartite graphs. The former statement was shown in [24] for $d = 1$ and in [10] for $d \geq 2$. The latter statement was shown in [18].

Let G be a connected bipartite graph. Let N be one of the two bipartition classes. Note that N is an independent set. Let F consist of all the edges between vertices of N. Then $G + F$ is a split graph. Hence, the theorem follows. □

The proof of our final result is similar to the proof of Lucke et al. [20] for showing that d-Cut is NP-complete for $3P_2$-free graphs for every $d \geq 2$.

Theorem 13. *For every $d \geq 2$, d-Cut is NP-complete on probe $4P_1$-free graphs.*

5 Conclusions

We precisely identified those graphs H, for which polynomial-time results for the problems d-Cut ($d \geq 1$), Perfect Matching Cut and Maximum Matching Cut can be extended from H-free graphs to probe H-free graphs. This yielded complete complexity dichotomies for all three problems on probe H-free graphs. Note such a dichotomy remains unknown for H-free graphs, with Theorems 1 and 2 still containing open cases.

We conclude that, despite our new polynomial-time results, these problems are harder for *probe* H-free graphs than for H-free graphs (if P $\neq$ NP) even for small graphs H. Moreover, we propose to study other graph problems on probe H-free graphs; so far, systematic studies have only been performed for Vertex Cover [5], Colouring [26] and the problems in this paper.

References

1. Ahn, J., Eagling-Vose, T., Lucke, F., Paulusma, D., Smith, S.: Finding d-cuts in claw-free graphs, Manuscript
2. Araújo, J., Cohen, N., Giroire, F., Havet, F.: Good edge-labelling of graphs. Discrete Applied Mathematics **160**, 2502–2513 (2012)
3. Berry, A., Golumbic, M.C., Lipshteyn, M.: Recognizing chordal probe graphs and cycle-bicolorable graphs. SIAM Journal on Discrete Mathematics **21**, 573–591 (2007)
4. Bonsma, P.S.: The complexity of the Matching-Cut problem for planar graphs and other graph classes. Journal of Graph Theory **62**, 109–126 (2009)
5. Brettell, N., Oostveen, J.J., Pandey, S., Paulusma, D., Rauch, J., van Leeuwen, E.J.: Computing subset vertex covers in H-free graphs. Theoretical Computer Science **1032**, 115088 (2025), conference version in Proc. FCT 2024
6. Chandler, D.B., Chang, M., Kloks, T., Liu, J., Peng, S.: On probe permutation graphs. Discrete Applied Mathematics **157**, 2611–2619 (2009)
7. Chang, M., Kloks, T., Kratsch, D., Liu, J., Peng, S.: On the recognition of probe graphs of some self-complementary classes of perfect graphs. Proc. COCOON 2005, Lecture Notes in Computer Science **3595**, 808–817 (2005)
8. Chvátal, V.: Recognizing decomposable graphs. Journal of Graph Theory **8**, 51–53 (1984)
9. Farley, A.M., Proskurowski, A.: Networks immune to isolated line failures. Networks **12**, 393–403 (1982)
10. Feghali, C., Lucke, F., Paulusma, D., Ries, B.: Matching cuts in graphs of high girth and H-free graphs. Proc. ISAAC 2023, LIPIcs **283**, 28:1–28:16 (2023)
11. Golovach, P.A., Paulusma, D., Song, J.: Computing vertex-surjective homomorphisms to partially reflexive trees. Theoretical Computer Science **457**, 86–100 (2012)

12. Golumbic, M.C., Lipshteyn, M.: Chordal probe graphs. Discrete Applied Mathematics **143**, 221–237 (2004)
13. Golumbic, M.C., Maffray, F., Morel, G.: A characterization of chain probe graphs. Annals of Operations Research **188**, 175–183 (2011)
14. Gomes, G., Sau, I.: Finding cuts of bounded degree: complexity, FPT and exact algorithms, and kernelization. Algorithmica **83**, 1677–1706 (2021)
15. Graham, R.L.: On primitive graphs and optimal vertex assignments. Annals of the New York Academy of Sciences **175**, 170–186 (1970)
16. Heggernes, P., Telle, J.A.: Partitioning graphs into generalized dominating sets. Nordic Journal of Computing **5**, 128–142 (1998)
17. Le, H., Le, V.B.: Complexity results for matching cut problems in graphs without long induced paths. Proc. WG 2023, LNCS **14093**, 417–431 (2023)
18. Le, V.B., Telle, J.A.: The Perfect Matching Cut problem revisited. Theoretical Computer Science **931**, 117–130 (2022)
19. Lucke, F., Marchand, J., Olbrich, J.: Finding minimum matching cuts in H-free graphs and graphs of bounded radius and diameter. CoRR **abs/2502.18942** (2025)
20. Lucke, F., Momeni, A., Paulusma, D., Smith, S.: Finding d-cuts in graphs of bounded diameter, graphs of bounded radius and H-free graphs. Proc. WG 2024, LNCS **14760**, 415–429 (2025)
21. Lucke, F., Paulusma, D., Ries, B.: On the complexity of Matching Cut for graphs of bounded radius and H-free graphs. Theoretical Computer Science **936**, 33–42 (2022)
22. Lucke, F., Paulusma, D., Ries, B.: Finding matching cuts in H-free graphs. Algorithmica **85**, 3290–3322 (2023)
23. Lucke, F., Paulusma, D., Ries, B.: Dichotomies for Maximum Matching Cut: H-freeness, bounded diameter, bounded radius. Theoretical Computer Science **1017**, 114795 (2024)
24. Moshi, A.M.: Matching cutsets in graphs. Journal of Graph Theory **13**, 527–536 (1989)
25. Patrignani, M., Pizzonia, M.: The complexity of the Matching-Cut problem. Proc. WG 2001, LNCS **2204**, 284–295 (2001)
26. Paulusma, D., Rauch, J., van Leeuwen, E.J.: Colouring probe H-free graphs. CoRR **abs/2505.20784** (2025)
27. Zhang, P., Schon, E.A., Fischer, S.G., Cayanis, E., Weiss, J., Kistler, S., Bourne, P.E.: An algorithm based on graph theory for the assembly of contigs in physical mapping of DNA. Computer Applications in the Biosciences **10**, 309–317 (1994)