An exact characterization of tractable demand patterns for maximum disjoint path problems

Dániel Marx[*] Paul Wollan[†]

Abstract
We study the following general disjoint paths problem: given a supply graph G, a set $T \subseteq V(G)$ of terminals, a demand graph H on the vertices T, and an integer k, the task is to find a set of k pairwise vertex-disjoint valid paths, where we say that a path of the supply graph G is valid if its endpoints are in T and adjacent in the demand graph H. For a class $\mathcal{H}$ of graphs, we denote by Maximum Disjoint $\mathcal{H}$-Paths the restriction of this problem when the demand graph H is assumed to be a member of $\mathcal{H}$. We study the fixed-parameter tractability of this family of problems, parameterized by k. Our main result is a complete characterization of the fixed-parameter tractable cases of Maximum Disjoint $\mathcal{H}$-Paths for every hereditary class $\mathcal{H}$ of graphs: it turns out that complexity depends on the existence of large induced matchings and large induced skew bicliques in the demand graph H (a skew biclique is a bipartite graph on vertices $a_1, \ldots, a_n, b_1, \ldots, b_n$ with a_i and b_j being adjacent if and only if $i \leq j$). Specifically, we prove the following classification for every hereditary class $\mathcal{H}$.

- If $\mathcal{H}$ does not contain every matching and does not contain every skew biclique, then Maximum Disjoint $\mathcal{H}$-Paths is FPT.
- If $\mathcal{H}$ does not contain every matching, but contains every skew biclique, then Maximum Disjoint $\mathcal{H}$-Paths is W[1]-hard, admits an FPT approximation, and the valid paths satisfy an analog of the Erdős-Pósa property.
- If $\mathcal{H}$ contains every matching, then Maximum Disjoint $\mathcal{H}$-Paths is W[1]-hard and the valid paths do not satisfy the analog of the Erdős-Pósa property.

1 Introduction

Given an undirected graph G and pairs of vertices $(s_1, t_1), \ldots, (s_k, t_k)$, the Disjoint Paths problem asks for pairwise vertex-disjoint paths $P_1, \ldots, P_k$ such that P_i has endpoints s_i and t_i. A celebrated result of Robertson and Seymour [28] (see also [12]) states that Disjoint Paths can be solved in time $f(k)n^3$ for some function f depending only on k, that is, there is a cubic-time algorithm for every fixed k. Therefore, Disjoint Paths is not only polynomial-time solvable for every fixed k, but fixed-parameter tractable parameterized by k. Recall that a problem is *fixed-parameter tractable* (FPT) parameterized by k if it can be solved in time $f(k)n^{O(1)}$ for some computable function f depending only on k.

Theorem 1.1. (Robertson and Seymour [28]) Disjoint Paths *can be solved in time* $f(k) \cdot n^{O(1)}$.

The main focus of the present paper is a natural maximization version of Disjoint Paths. Given an undirected graph G, pairs of vertices $(s_1, t_1), \ldots, (s_m, t_m)$, and an integer k, the Maximum Disjoint Paths problem asks for a set of k pairwise vertex-disjoint *valid* paths, where we say that a path is valid if its endpoints are s_j and t_j for some $1 \leq j \leq m$. We will typically refer to the graph G as the *supply graph* and the graph with vertex set $\{s_1, \ldots, s_m, t_1, \ldots, t_m\}$ and edge set s_it_i for $1 \leq i \leq m$ as the *demand graph.* The Maximum Disjoint Paths problem remains NP-complete even with strong restrictions on the input: it is NP-complete when restricted to problem instances with supply graph G and demand graph H such that $G \cup H$ is planar [20]. See [22] for an in depth discussion of variants of the problem that are known to be computationally hard, as well as [13] for surveys on the problem.

In contrast, for every fixed k it is easy to see that Maximum Disjoint Paths is polynomial-time solvable: we guess k integers $1 \leq j_1 < j_2 < \cdots < j_k \leq m$, and then solve the Disjoint Paths instance on G with pairs $(s_{j_1}, t_{j_2}), \ldots, (s_{j_k}, t_{j_k})$ using the algorithm of Theorem 1.1. Clearly, the Maximum Disjoint Paths instance has a solution if and only if at least one of the instances of Disjoint Paths has. As there are $m^{O(k)}$ different ways of selecting the k integers $j_1, \ldots, j_k$, this results in an $f(k)n^{O(k)}$ time algorithm. But is Maximum Disjoint Paths fixed-parameter tractable? As we shall see later in this paper, Maximum Disjoint Paths is W[1]-hard, which means that it is not FPT under standard complexity assumptions. The hardness result holds even if G is a planar graph whose treewidth is bounded by

[*]Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI) dmarx@cs.bme.hu. Research supported by the European Research Council (ERC) grant "PARAMTIGHT: Parameterized complexity and the search for tight complexity results," reference 280152 and OTKA grant NK105645.

[†]Department of Computer Science, University of Rome, wollan@di.uniroma1.it. Research supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC Grant Agreement no. 279558.

a function of k. This indicates that two fundamental algorithmic ideas underlying the Disjoint Paths algorithm of Robertson and Seymour [28] cannot be used for Maximum Disjoint Paths: finding irrelevant vertices exploiting properties of graphs embedded on surfaces (or excluding minors) and using dynamic programming to solve bounded-treewidth instances.

Despite the hardness of the general problem, there are easier special cases of Maximum Disjoint Paths: classic results yield polynomial time algorithms even with k as part of the input when the problem is restricted to certain types of demand graphs. Suppose that S and T are two sets of vertices and the set of pairs given in the input is $S \times T$ (that is, every pair (s, t) with $s \in S$, $t \in T$ is listed in the input; note that the problem definition does not require the pairs to be disjoint). Then the valid paths are the paths connecting S and T, hence it can be checked in polynomial time if there are k valid paths by solving a maximum flow problem with vertex capacities. The demand graphs in this case are complete bipartite graphs. A result of Mader [15] generalizes this observation by giving a min-max theorem for the maximum number of disjoint valid paths when the demand graph is a multi-partite graph. Mader's theorem is existential, but a maximal set of disjoint valid paths can be algorithmically found in polynomial time as an application of Lovász' matroid matching algorithm [14]. In a recent paper, Hirai and Pap [10] exactly characterized which demand graphs make a more general version of the *weighted edge-disjoint paths* problem polynomial time solvable.

It is possible to use the Robertson-Seymour algorithm for the Disjoint Paths problem to find instances that are FPT parameterized by the number k of paths, but NP-complete when k is part of the input. Consider for example the case when the set of pairs is $(S_1 \times T_1) \cup (S_2 \times T_2)$ for pairwise disjoint subsets $S_1, S_2, T_1, T_2 \subseteq V(G)$. This case of the problem is a restatement of the node-capacitated 2-commodity flow problem and is NP-complete when k is included in the input [5]. To show that this case is FPT, we can proceed in the following way. First, we guess the number $0 \leq k_1 \leq k$ of paths in the solution that connect S_1 and T_1 (and hence $k_2 = k - k_1$ paths connect S_2 and T_2). Let us introduce k_1 vertices $s^1_1, \ldots, s^1_{k_1}$, all of them fully connected to S_1; another k_1 vertices $t^1_1, \ldots, t^1_{k_1}$, all of them fully connected to T_1. Similarly, we introduce k_2 vertices $s^2_1, \ldots, s^2_{k_2}$ fully connected to S_1 and k_2 vertices $t^2_1, \ldots, t^2_{k_2}$ fully connected to T_2. Then the required $k_1 + k_2$ paths exist if the Disjoint Paths instance with pairs $(s^1_1, t^1_1), \ldots, (s^1_{k_1}, t^1_{k_1}), (s^2_1, t^2_1), \ldots, (s^2_{k_2}, t^2_{k_2})$ has a solution. Therefore, we can reduce the problem to $k + 1$ instances of Disjoint Paths, implying that this special case of Maximum Disjoint Paths is FPT.

Our main goal is to understand which demand patterns make Maximum Disjoint Paths fixed-parameter tractable. The formal setting of our investigations is the following. First, we introduce a slightly different formulation of Maximum Disjoint Paths. Let G be the supply graph, $T \subseteq V(G)$ be a set of terminals, and H be the demand graph defined on the vertices T. We say that a path in G is *valid* if both of its endpoints are in T and they are adjacent in H. The task is now to find k pairwise vertex-disjoint valid paths. The examples above can be expressed by an instance where H is a biclique (complete bipartite graph) or the disjoint union of two bicliques. For a class $\mathcal{H}$ of graphs, we define Maximum Disjoint $\mathcal{H}$-Paths as the special case Maximum Disjoint Paths when H is restricted to be a member of $\mathcal{H}$.

For example, as we have seen, if $\mathcal{H}$ is the class of all bicliques, then Maximum Disjoint $\mathcal{H}$-Paths is polynomial-time solvable and if every graph in $\mathcal{H}$ is the disjoint union of two bicliques, then Maximum Disjoint $\mathcal{H}$-Paths is fixed-parameter tractable. One can observe that the argument can be generalized to the case when the two bicliques are not disjoint (i.e., the demand graph H graph is obtained by fully connecting S_1 with T_1 and S_2 with T_2, where these four sets are not necessarily disjoint), or to the case where every graph in $\mathcal{H}$ is the (not necessarily disjoint) union of c bicliques for some constant c, or to the case where every graph $H \in \mathcal{H}$ has the property that the vertices in H have at most c different neighborhoods for some constant c. Therefore, there are fairly complicated demand patterns that make the problem FPT.

Formally, our goal is to identify every class $\mathcal{H}$ for which Maximum Disjoint $\mathcal{H}$-Paths is FPT. For technical reasons, we restrict our attention to classes $\mathcal{H}$ that are *hereditary,* that is, closed under taking induced subgraphs. Intuitively, if H' is an induced subgraph of some $H \in \mathcal{H}$, then adding H' to $\mathcal{H}$ should not make the problem any harder: given an instance with demand pattern H', we can easily express it with demand pattern H by introducing dummy isolated terminals into the supply graph G to represent the vertices $V(H) \setminus V(H')$. Therefore, it seems justified to study only graph classes that are closed under taking induced subgraphs. However, there is no formal reduction showing that if every graph in $\mathcal{H}'$ is an induced subgraph of a member of $\mathcal{H}$, then the fixed-parameter tractability of the problem with $\mathcal{H}$ implies the fixed-parameter tractability of the problem with $\mathcal{H}'$. There are at least two technical issues with the simple reduction described above: first, adding the isolated vertices may increase the size of the instance if H is much larger than H' and, second, even if we know that $H' \in \mathcal{H}'$ is a subgraph of some $H \in \mathcal{H}$, finding such an H may be computationally hard. Therefore, to avoid the discussion of artificial technicalities, we consider only hereditary classes.

Our results. First, we investigate a purely combinatorial question. A classical result of Erdős and Pósa [4] states that in every undirected graph G, the minimum number of vertices needed to cover every cycle in G can be bounded by a function of the maximum number of vertex-disjoint cycles. This result motivates the following definition: we say that a set $\mathcal{C}$ of graphs has the *Erdős-Pósa property* if there is a function $f(k)$ such that every graph G has either k vertex-disjoint subgraphs that belong to $\mathcal{C}$ or a set X of at most $f(k)$ vertices such that $G - X$ has no subgraph that belongs to $\mathcal{C}$; the result of Erdős and Pósa [4] can be stated as saying that the set of all cycles has this property. The literature contains numerous results proving that the Erdős-Pósa property holds for variants of the disjoint cycle problem such as disjoint long cycles [1], directed cycles [25], cycles of length 0 mod m [30], as well as characterizing when the Erdős-Pósa property holds for odd cycles [24, 31, 23, 11] and cycles of non-zero length mod m [32]. Further study has considered whether sets $\mathcal{C}$ defined by other containment relations such as minors also have the Erdős-Pósa property [26, 3].

We investigate the natural analog of the Erdős-Pósa property in the context of the Maximum Disjoint Paths problem: Is it true that the valid paths have the Erdős-Pósa property, that is, is it true that either there are k valid paths or a set of at most $f(k)$ vertices covering every valid path? Besides its combinatorial interest, we explore this question because the Erdős-Pósa property of some objects is often correlated with good algorithmic behavior of the corresponding packing/covering problems, especially from the viewpoint of fixed-parameter tractability. However, in general, the answer to this question is no. The standard counterexample is an $n \times n$ grid graph with the vertices $s_1, \ldots, s_n$ appearing in the top row from left to right, and the vertices $t_1, \ldots, t_n$ appearing in the bottom row from right to left. Then every $s_i - t_i$ path intersects every $s_j - t_j$ path for $i \neq j$, but we need $n - 1$ vertices to cover all such paths. Therefore, the Erdős-Pósa property does not hold for valid paths in general, but may hold for the Maximum Disjoint $\mathcal{H}$-Paths problem for certain (hereditary) classes $\mathcal{H}$. For example, if $\mathcal{H}$ contains only bicliques, then Menger's Theorem states that the Erdős-Pósa property holds in a tight way with $f(k) = k - 1$; if $\mathcal{H}$ contains only cliques, then a classical result of Gallai [8] states that the the Erdős-Pósa property holds with $f(k) = 2k - 2$.

Let M_r be the graph consisting of a matching of size r (i.e., M_r has $2r$ vertices and r edges). The counterexample above shows that if the hereditary class $\mathcal{H}$ contains M_r for every $r \geq 1$, then the Erdős-Pósa property surely does not hold. Surprisingly, this is the only obstacle: our first result states that if $\mathcal{H}$ is a hereditary class of graphs not containing M_r for every $r \geq 1$, then the valid paths in Maximum Disjoint $\mathcal{H}$-Paths have the Erdős-Pósa property. Our proof is algorithmic and gives an algorithm that either produces a set of disjoint valid paths or a hitting set Z covering every valid path.

Theorem 1.2. *Let $\mathcal{H}$ be a hereditary class of graphs, and assume there exists an integer $r \geq 1$ such that $M_r \notin \mathcal{H}$. There exists an algorithm which given a graph G, $T \subseteq V(G)$, integer $k \geq 1$, and $H \in \mathcal{H}$ with $V(H) = T$, returns one of the following:*

1. *a set of k pairwise disjoint valid paths, or*
2. *a set Z of at most $2^{O(k+r)}$ vertices such that every valid path intersects Z.*

Moreover, the algorithm runs in time $2^{2^{O(k+r)}}(|V(G)| + |E(G)|)^{O(1)}$.

By a well-known observation (cf. [17]), the algorithm of Theorem 1.2 can be turned into an FPT approximation algorithm of the following form.

Corollary 1.1. *Let $\mathcal{H}$ be a set of graphs closed under taking induced subgraphs, and assume there is an integer $r \geq 1$ such that $M_r \notin \mathcal{H}$. Then there is a polynomial-time algorithm that, given an instance of* Maximum Disjoint $\mathcal{H}$-Paths*, finds a solution with $\Omega(\log \log OPT)$ disjoint valid paths, where OPT is the maximum size of a set of pairwise disjoint valid paths.*

Can we improve the algorithm of Theorem 1.2 to an exact FPT algorithm that either finds a set of k disjoint valid paths or correctly states that there is no such set? It seems that we need one more property of $\mathcal{H}$ for the existence of such algorithms. A *skew biclique of size $n + n$* is the bipartite graph S_n on vertices $a_1, \ldots, a_n, b_1, \ldots, b_n$ such that a_i and b_j are adjacent if and only if $i \leq j$. Even though the (hereditary closure of) the set $\mathcal{H}$ of all skew bicliques has the Erdős-Pósa property by Theorem 1.2 (as skew bicliques do not have large induced matchings), disjoint paths problems with skew biclique demand patterns can be hard. Our main result states that large induced matchings and large skew bicliques are the only demand patterns that make the Maximum Disjoint $\mathcal{H}$-Paths problem hard.

Theorem 1.3. *Let $\mathcal{H}$ be a hereditary set of graphs. If there is an integer $r \geq 1$ such that $M_r, S_r \notin \mathcal{H}$, then* Maximum Disjoint $\mathcal{H}$-Paths *is* FPT*; otherwise,* Maximum Disjoint $\mathcal{H}$-Paths *is* W[1]*-hard.*

Therefore, we have obtained a tight characterization of the fixed-parameter tractable cases of Maximum Disjoint $\mathcal{H}$-Paths. Observe that the algorithmic part of Theorem 1.3 covers the FPT cases we discussed above: if the vertices in every $H \in \mathcal{H}$ have at most c different neighborhoods, then $\mathcal{H}$ cannot contain every matching and every skew biclique. However, Theorem 1.3 gives some more general FPT cases as well: for example, if every graph in $\mathcal{H}$ is a biclique minus a matching of arbitrary size, then clearly there are no large induced matchings or skew bicliques in $\mathcal{H}$, but the number of different neighborhoods can be arbitrarily large. Observe also that Corollary 1.1 and Theorem 1.3 exhibit a large class of problems that are W[1]-hard, but admit an FPT approximation: if $\mathcal{H}$ contains every skew biclique S_r, but does not contain some matching M_r, then Maximum Disjoint $\mathcal{H}$-Paths is such a problem. There is only a handful of known problems with this property (see [17, 9, 2]), thus this may be of independent interest.

Our techniques. The first observation in the proof of Theorem 1.2 is that if there is a small set Z of vertices such that more than one component of $G - Z$ contains valid paths, then we can solve the problem recursively. Therefore, we may assume that the valid paths are quite intertwined, giving us a notion of connectivity similar to tangles. Our first goal is to find a certain number of pairs (s_1, t_1), ..., (s_h, t_h) such that s_i and t_i are adjacent in H and the set $\{s_1, \ldots, s_h, t_1, \ldots, t_h\}$ is highly connected in our notion of connectivity. In particular, the connectivity ensures that there are many disjoint paths between $\{s_1, \ldots, s_h\}$ and $\{t_1, \ldots, t_h\}$. This is not quite what we need: all we know is that s_i and t_i are adjacent in H, but we have no information about the adjacency of s_i and t_j for $i \neq j$. This is the point where we exploit the assumption that there are no large induced matchings in H. A simple Ramsey-type argument shows that if a graph has a large (not necessarily induced) matching, then it either has a large induced matching or a large biclique. By assumption, there is no large induced matchings in H, which means that H contains a large biclique on the vertices $\{s_1, \ldots, s_h, t_1, \ldots, t_h\}$. Then by the connectivity of this set, we can realize k disjoint paths with endpoints in this biclique.

The fixed-parameter tractability part of Theorem 1.3 is proved the following way. First, we bootstrap the algorithm with the approximation of Theorem 1.2: we obtain either k disjoint valid paths (in which case we are done) or a set Z of bounded size covering every valid path. In the latter case, we solve the problem by analyzing the components of $G-Z$: as there are no valid paths in any component C of $G-Z$, essentially what we need to understand is how subsets of terminals in C can be connected to Z. However, each component of $G - Z$ can contain a large number of terminals and there can be a large number of components of $G - Z$. First, in each component C of $G - Z$, we reduce the number of terminals so that their number is bounded: we identify terminals that are *irrelevant,* that is, we can prove that if there is a solution, then there is a solution not using these terminals. To identify irrelevant terminals, we use the concept of *representative sets,* which were already used in the design of FPT algorithms, mostly for path and matroid problems [21, 18, 6, 7, 29]. While the concept is the same as in previous work, the reason why we can give a bound on the size of representative sets is very different: as shown by a simple Ramsey-type argument, it is precisely the lack of large induced matchings and skew bicliques in $\mathcal{H}$ that makes the argument work. (More precisely, we need to exclude large cliques as well, but we have a separate argument for that.) Our algorithm can be seen as a generalization of the ideas in the data structure of Monien [21], but it does not use any of the more advanced matroid-based techniques of more recent work [18, 6, 7, 29]. After reducing the number of terminals to a constant in each component of $G - Z$, next we use elementary arguments to show that every terminal in all but a bounded number of components is irrelevant. Thus we have a bound on the total number of terminals and then we can use the algorithm of Robertson and Seymour [28] on every set of k pairs of terminals.

The hardness part of Theorem 1.3 states W[1]-hardness for infinitely many classes $\mathcal{H}$. However, we need to prove only the following two concrete W[1]-hardness results: when the pattern is a matching and when the pattern is a skew biclique. We prove these hardness result in a slightly stronger form: the supply graph G is restricted to be planar and we show that the problems are hard even when parameterized by both the number of paths k to be found and the treewidth w of the supply graph, that is, even an algorithm with running time $f(k, w) \cdot n^{O(1)}$ seems unlikely.

Theorem 1.4. *If $\mathcal{H}$ contains M_r for every $r \geq 1$, then* Maximum Disjoint $\mathcal{H}$-Paths *is* W[1]*-hard with combined parameters k and w (where w is the* *treewidth of G), even when restricted to instances where G is planar.*

Theorem 1.5. *If $\mathcal{H}$ contains S_r for every $r \geq 1$, then* Maximum Disjoint $\mathcal{H}$-Paths *is* W[1]*-hard with combined parameters k and w (where w is the treewidth of G), even when restricted to instances where G is planar.*

These hardness proofs will appear in the full version of the paper. Note that Theorem 1.5 actually implies Theorem 1.4: if $\mathcal{H}$ contains the matching M_r for every $r \geq 1$, then it is easy to simulate any demand pattern, including skew bicliques. The reduction is as follows. First, if vertex v has degree d in H, then let us attach d degree-1 neighbors to v and make them terminals. Then replace each edge (x, y) of H with an edge connecting a degree-1 neighbor of x and a degree-1 neighbor of y not incident to any demand edge yet. This way the new demand graph becomes a matching of $|E(H)|$ edges. Therefore, giving a separate proof for Theorem 1.4 is redundant. Nevertheless, we give a self-contained W[1]-hardness proof of Maximum Disjoint Paths with no restriction on the demand pattern, which, by the reduction described above, proves Theorem 1.4 (but not Theorem 1.5). We believe that the W[1]-hardness of Maximum Disjoint Paths can be already of independent interest and the proof is much simpler and cleaner than the highly technical proof of Theorem 1.5.

2 Excluding induced matchings: Erdős-Pósa property and FPT approximation

In this section, we give the proof of Theorem 1.2. We begin with a more technical statement which will facilitate the recursive step of the algorithm.

Theorem 2.1. *Let G be a graph, $T \subseteq V(G)$, $k, r \geq 1$ integers, and H a graph with $V(H) = T$. Assume that T is an independent set and $\deg_G(v) = 1$ for all $v \in T$. There exists an algorithm which takes as input G, T, k, r, and H and returns one of the following:*

1. *k pairwise disjoint valid paths, or*
2. *a set X of at most $4 \cdot 5^{20(k+r)}$ vertices such that every valid path intersects X.*
3. *a subset $Z \subseteq T$ with $|Z| = 2r$ such that $H[Z]$ is an induced matching.*

Moreover, the algorithm runs in time $4^{3 \cdot 5^{10(k+r)}}(|V(G) \setminus T| + |E(G)|)^{O(1)}$.

Theorem 1.2 follows easily from Theorem 2.1. The proof of Theorem 2.1 will occupy the remainder of the section; we outline how the proof will proceed. Consider for a moment a more general problem. Assume we are trying to show that the Erdős-Pósa property holds for a set $\mathcal{C}$ of connected graphs: i.e. that there exists a function f such that for every positive integer k and graph G, either G has k disjoint subgraphs in $\mathcal{C}$ or there exists $f(k)$ vertices intersecting every subgraph of G in $\mathcal{C}$. If we consider a minimal counterexample, then there cannot exist a separation (X, Y) of small order such that each of X and Y contain a subgraph in $\mathcal{C}$. Otherwise, by minimality, we can either find $k-1$ disjoint $\mathcal{C}$ subgraphs in $X - Y$ or a set of $f(k-1)$ vertices in $X - Y$ intersecting all such subgraphs. If we found $k-1$ subgraphs, along with the graph in Y, we would have k subgraphs in $\mathcal{C}$, contradicting our choice of counterexample. Thus, we may assume there is hitting set Z_X of size $f(k-1)$ intersecting every $\mathcal{C}$ subgraph in $X - Y$. Similarly, there exists a bounded hitting set Z_Y in $Y - X$. By our assumption that $\mathcal{C}$ consists of only connected subgraphs, every subgraph of G in $\mathcal{C}$ must be contained in either X or Y. Thus, $Z_X \cup Z_Y \cup V(X \cap Y)$ is a hitting set of all $\mathcal{C}$ subgraphs in G of size $2f(k-1)+|X \cap Y|$. If the function f grows sufficiently quickly, this will yield a contradiction.

The conclusion is that for every small order separation (X, Y), only one of X or Y can contain a subgraph in $\mathcal{C}$. This defines a *tangle* in the graph G. Tangles are a central concept in the Robertson-Seymour theory of graph minors [27]. We will not need the exact definitions here, as we do not use any technical tangle results. However, this argument shows how tangles arise naturally in proving Erdős-Pósa type results; see [32] for another example. The proof of Theorem 2.1 is not presented in terms of tangles for two reasons. First, the tangle defined above only exists in a minimal counterexample to the theorem. While this suffices for an existential proof of an Erdős-Pósa bound, we are also interested in an algorithm. We need to consider all possible problem instances and then we will not always have such a tangle to work with. Second, the proof does not use any technical tangle theorems; in the interest of simplicity of the presentation, we do not introduce tangles although they inform and motivate how the proof proceeds.

Now return to the specific problem at hand. Consider a graph G, k, r, $T \subseteq V(G)$, and demand graph H with $V(H) = T$. A subset $X \subseteq T$ is *well-linked* if for any $U, W \subseteq X$ with $|U| = |W|$, there exist $|U|$ disjoint paths from U to W. We attempt to find a large subset $T' \subseteq T$ such that

1. $H[T']$ contains a perfect matching, and
2. T' is well-linked in G.

Given a large set T' satisfying 1 and 2, the argument is fairly straightforward. By a simple Ramsey-type argument, either $H[T']$ contains an induced matching of size r or there exist two sets U and W in T', each of size k, such that every vertex in U is adjacent every vertex in W (in H). As we are assuming T' is well-linked in G, given such a U and W, we can find k disjoint paths from U to W and these will necessarily be valid paths.

How can we find such a subset T' of T? It is easy to find such a subset T' of size two — take two vertices in T which are adjacent in H and connected by a path. Thus, the difficulty will lay in showing we can find T' sufficiently large to apply the desired Ramsey-type argument. The property of being well-linked is a standard certificate that a graph has a large tangle. We proceed by effectively showing that we either have a tangle as in a minimal counterexample to the Erdős-Pósa property as in the above paragraph, or alternatively, finding a separation separating two valid paths and then recurse on the smaller graphs. More explicitly, we replace property 2 above by:

$2'$. there does not exist a separation (X, Y) of $G - (T \setminus T')$ of order $< |T'|$ with $T' \subseteq V(X)$ and Y containing a valid path.

Property $2'$ forces a similar behavior to well-linkedness in a tangle without requiring the technical properties of a tangle. We show that either we can grow T' by two vertices and satisfy 1 and $2'$, or alternatively find a separation where we can recurse. We proceed by showing several technical lemmas in Subsections 2.1 and 2.2 in preparation for the proof of Theorem 2.1 which follows in Subsection 2.3.

2.1 $\mathcal{P}$-tight separations

We begin by introducing what we will call tight separations.

Definition 1. ($\mathcal{P}$-tight) *Let G be a graph and $T \subseteq V(G)$. Let $\mathcal{P}$ be a set of connected subgraphs in $G - T$. We say a separation (U, W) is* $\mathcal{P}$-tight *for the pair (G, T) if*

i. $T \subseteq V(U)$;
ii. *there exists $P \in \mathcal{P}$ with $P \subseteq W - U$;*
iii. *there does not exist a separation (U', W') and element $P \in \mathcal{P}$ with $|U' \cap W'| \leq |U \cap W|$, $U \subsetneq U'$, and $P \subseteq W' - U'$.*

When there can be no confusion as to the set $\mathcal{P}$, we will simply say a separation is tight *for the pair (G, T).*

Thus a separation is tight if the portion not containing T is made as small as possible while not increasing the order of the separation and maintaining the property that it still contains an element of $\mathcal{P}$. Note that a tight separation may have order greater than $|T|$. However, a tight separation of order at most $|T|$ always exists if $\mathcal{P}$ is non-empty. To see this, let (U, W) to be a separation of minimum order satisfying i and ii, and subject to that, to maximize $|V(U)| + |E(U)|$. Such a separation always exists as the trivial separation (T, G) satisfies i and ii with T treated as the graph with vertex set T and no edges. Then (U, W) will be of order at most $|T|$ and satisfy $i - iii$. A similar argument yields the following observation.

Observation 1. *Let G be a graph, $T \subseteq V(G)$, and $\mathcal{P}$ a non-empty set of connected graphs in $G - T$. Let (U, W) be a separation satisfying i and ii in the definition of tight for the pair (G, T). There exists a separation (U', W') of order at most $|U \cap W|$ which is tight for (G, T) and $U \subseteq U'$. Let $T' \subseteq T$. If (U, W) is a $\mathcal{P}$-tight separation for the pair (G, T), then $(U - (T \setminus T'), W - (T \setminus T'))$ is a $\mathcal{P}$-tight separation for the pair $(G - (T \setminus T'), T')$.*

Let G be a graph, $T \subseteq V(G)$, and $\mathcal{P}$ a set of connected subgraphs in $G - T$. We say that the set T is *$\mathcal{P}$-free* if there does not exist a separation (U, W) of order strictly less than $|T|$ and $P \in \mathcal{P}$ such that $T \subseteq U$ and $V(P) \subseteq W - U$.

Lemma 2.1. *Let G be a graph, $T, T' \subseteq V(G)$ with $T' \subseteq T$ and let $G' = G - (T \setminus T')$. Let $\mathcal{P}$ a set of connected subgraphs in $G - T$. Assume that T' is $\mathcal{P}$-free in G'. Let $t \geq 1$ be a positive integer. Let (U', W') be a $\mathcal{P}$-tight separation of the pair (G', T') of order t, and let (U_1, W_1) and (U_2, W_2) be distinct $\mathcal{P}$-tight separations of the pair (G, T), each of order $t + 1$. Then one of the following holds:*

1. *$V(U' \cap W') \cup V(U_1 \cap W_1) \cup V(U_2 \cap W_2)$ is a hitting set for $\mathcal{P}$.*
2. *There exists $P \in \mathcal{P}$ such that P is contained in one of the graphs U', U_1, or U_2.*
3. *$V(U_1) \cap V(U_2) = V(U') \cup T$.*

Proof. We may assume there exists $P \in \mathcal{P}$ which is disjoint from the set $V(U' \cap W') \cup V(U_1 \cap W_1) \cup V(U_2 \cap W_2)$. Lest we satisfy 2, we may assume as well that $P \subseteq W_1 \cap W_2 \cap W'$. Note by construction that P is disjoint from $U_1 \cup U_2 \cup U'$.

Fix $i \in \{1, 2\}$. We first show that $U' \subseteq U_i$. We would like to consider the two pairs $(U' \cap U_i, W' \cup W_i)$ and $(U' \cup U_i, W' \cap W_i)$ with the first being a separation of G' and the second being a separation of G. However, there is a slight technicality, namely possible vertices in the set $(T \setminus T') \cap V(W_i)$. Fix $X = (T \setminus T') \cap V(W_i)$. The pair $(U' \cap U_i, W' \cup W_i)$ may not be a separation of G' as there may be vertices of $T \setminus T'$ in W_i, however $(U' \cap U_i, (W' \cup W_i) - X)$ is a separation of G'. Similarly, $(U' \cup U_i, W' \cap W_i)$ may not be a separation of G as vertices in X may have neighbors in $(W' \cap W_i) - (U' \cup U_i)$. Let $(W' \cap W_i) + X$ be the subgraph of G formed by $W' \cap W_i$ along with the vertex set X and any edge of G with an end in X and an end in $W' \cap W_i$. It follows that $(U' \cup U_i, (W' \cap W_i) + X)$ is a separation of G. A counting argument shows that the sum of the orders of the separations $(U' \cap U_i, (W' \cup W_i) - X)$ and $(U' \cup U_i, (W' \cap W_i) + X)$ is equal to the sum of the orders of the two separations (U', W') and (U_i, W_i), namely $2t + 1$.

The separation $(U' \cap U_i, (W' \cup W_i) - X)$ is a separation of G' with $T' \subseteq V(U' \cap U_i)$. Moreover, the path P is contained in $((W' \cup W_i) - X) - (U' \cap U_i)$. We conclude from the fact that T' is $\mathcal{P}$-free that the separation has order at least t.

It follows that $(U' \cup U_i, (W' \cap W_i) + X)$ is a separation of G of order at most $t + 1$ with $P \subseteq ((W' \cap W_i) + X) - (U' \cup U_i)$. It follows that $U' \cup U_i = U_i$ by property *iii* in the definition of tight for (U_i, W_i) and thus $U' \subseteq U_i$ as desired.

We conclude that $V(U') \cup T \subseteq V(U_1) \cap V(U_2)$. The separation $(U_1 \cup U_2, W_1 \cap W_2)$ must be of order at least $t+2$, lest we violate *iii* for one of the separations (U_1, W_1) or (U_2, W_2). Note that here we are using the fact that the separations (U_1, W_1) and (U_2, W_2) are distinct. It follows that $(U_1 \cap U_2, W_1 \cup W_2)$ is a separation of order at most t. Consequently, $((U_1 \cap U_2) - (T \setminus T'), (W_1 \cup W_2) - (T \setminus T'))$ is a separation of order at most t of the graph G' with $V(U') \subseteq V(U_1 \cap U_2) - (T \setminus T')$. By *iii* in the definition of tight for (U', W'), we have that $V(U') = V(U_1 \cap U_2) - (T \setminus T')$, completing the proof. □

2.2 Algorithms for $\mathcal{P}$-free sets and $\mathcal{P}$-tight separations In this section, we consider the algorithmic problem of finding tight separations. If we were given $\mathcal{P}$ as a list of subgraphs, one could use standard flow algorithms to find a minimum order separation separating the terminals T from each element of $\mathcal{P}$. A suitably minimal separator will determine if T is $\mathcal{P}$-free. However, in the applications to come, the we will not have any reasonable bound on the size of $\mathcal{P}$. Thus, we assume $\mathcal{P}$ is given by an oracle and bound the runtime in the size of the terminal set T. The algorithms will be based on the idea of finding important separators.

Definition 2. (separator) *Let G be an undirected graph and let $X, Y \subseteq V(G)$ be two disjoint sets. A set $S \subseteq V(G)$ of vertices is an $X - Y$* separator *if S is disjoint from $X \cup Y$ and there is no component K of $G - S$ with both $V(K) \cap X \neq \emptyset$ and $V(K) \cap Y \neq \emptyset$.*

Definition 3. (important separators) *Let $X, Y \subseteq V(G)$ be disjoint sets of vertices, $S \subseteq V(G)$ be an $X - Y$ separator, and let K be the union of the vertex sets of every component of $G - S$ intersecting X. We say that S is an important $X - Y$ separator if it is inclusionwise minimal and there is no $X - Y$ separator S' with $|S'| \leq |S|$ such that $K' \supsetneq K$, where K' is the union of every component of $G - S'$ intersecting X.*

Lemma 2.2. ([19]) *Let $X, Y \subseteq V(G)$ be disjoint sets of vertices in a graph G. For every $p \geq 0$, there are at most 4^p important $X - Y$ separators of size at most p. Furthermore, we can enumerate all these separators in time $4^p \cdot p \cdot (|E(G)| + |V(G)|)$.*

Let G be a graph, $T \subseteq V(G)$, and $\mathcal{P}$ a set of connected subgraphs of $G - T$. We will show that there is an algorithm for efficiently testing whether T is $\mathcal{P}$-free or not for sets T of bounded size. In the applications to come, the set $\mathcal{P}$ will typically have a super-polynomial number of elements. Thus, we will typically assume that $\mathcal{P}$ is given by an oracle. A *$\mathcal{P}$-oracle* is a function f such that for any subgraph $H \subseteq G$, f responds "yes" if there is an element $P \in \mathcal{P}$ such that $P \subseteq H$ and "no" otherwise. A *certificate* that T is not free is a separation (X, Y) of order strictly less than $|T|$ such that $T \subseteq V(X)$ and there exists $P \in \mathcal{P}$ with $P \subseteq Y - X$.

Test $\mathcal{P}$-Free
Input: A graph G, $T \subseteq V(G)$, $\mathcal{P}$-oracle f for a set $\mathcal{P}$ of connected subgraphs of $G - T$.
Find: either

- confirm that T is $\mathcal{P}$-free or
- output a separation (X, Y) which is a certificate that T is not free; moreover, (X, Y) is of minimum order among all such separations.

Lemma 2.3. (Testing if a separation is $\mathcal{P}$-free) *There exists an algorithm solving* Test $\mathcal{P}$-Free *running in time $4^{|T|}|V(G)|^{O(1)}$ utilizing $O(|V(G)|4^{|T|})$ calls of the $\mathcal{P}$-oracle.*

Proof. Let G, T, and an oracle f for the set $\mathcal{P}$ be given. Let $n = |V(G)|$ and $m = |E(G)|$. There is a slight technical issue which we must address. We want to proceed by calculating all separations (X, Y) where $T \subseteq X$ and $X \cap Y$ is an important separator for some vertex $y \in Y$. However, we will additionally need to consider such separations where $X \cap Y$ intersects the set T. However, in the definition of important separator, we do not consider separators which intersect one of the two sets. Thus, we define an auxiliary graph G' formed by adding a new vertex a' adjacent to every vertex of T and consider important separators separating a vertex from a' in G'.

Fix a vertex $y \in V(G) \setminus T$. Enumerate all important y–a' separators in G' of size at most $|T|-1$. For each separator S, let K_S be the component of $G' - S$ containing y. Using the $\mathcal{P}$-oracle f, check if there exists an element $P \in \mathcal{P}$ with $P \subseteq K_S$. We do this for every $y \in V(G) \setminus T$. By Lemma 2.2, this can be done in time $O(4^{|T|}|T|(n+m)n \cdot m)$ with at most $n4^{|T|}$ calls to to the $\mathcal{P}$-oracle, as desired.

Assume, as a case, we find a vertex $y \in V(G) \setminus T$ and an important $y - a'$ separator S such that the subgraph induced by K_S contains an element of $\mathcal{P}$. Pick y and S over all such vertices and important separators to minimize $|S|$. We return the separation $((G-V(K_S))-E(G[S]), G[V(K_S)\cup S])$ as a certificate that T is not $\mathcal{P}$-free. If we find no such important separator, we return that T is $\mathcal{P}$-free.

To see correctness, first observe that if we return a separation, it must be the case that T is not $\mathcal{P}$-free. Thus, we must only show that if T is not $\mathcal{P}$-free, we correctly find a minimum order separation certifying so. Assume that T is not free, and let (X,Y) be a separation such that:

i. $T \subseteq V(X)$ and there exists $P \in \mathcal{P}$ such that $P \subseteq Y - X$.
ii. Subject to i, the size of $|X \cap Y|$ is minimized.
iii. Subject to i and ii, $|Y|$ is minimized.

Moreover, assume that the algorithm finds no important separator S such of order at most $|X \cap Y|$ such that S separates a' from an element of $\mathcal{P}$. Note, by iii, we may assume that $Y - X$ is connected.

Let y be a vertex of $Y-X$. The set $X\cap Y$ is an y–a' separator in G'. If $X\cap Y$ is an important separator, then we returned the separation $((G-V(K_S))-E(G[S]), G[V(K_S)\cup S])$ for some important separator S. Thus, we correctly identified that T is not free. Moreover, it must be the case that $|S| \leq |X\cap Y|$, and therefore, $((G-V(K_S))-E(G[S]), G[V(K_S)\cup S])$ is a minimum order separation certifying that T is not free, contrary to our assumption.

Thus, we have that $X \cap Y$ is not an important $y - a'$ separator. By our choice of (X,Y) to satisfy ii, we have that $X\cap Y$ is a minimal (by containment) $y - a'$ separator. We conclude that there exists an important y–a' separator S of order $|X\cap Y|$ such that the component of $G' - S$ containing y contains all of $Y-X$. Note that here we are using the fact that $Y-X$ is connected. Thus, the separator S separates a' from an element of $\mathcal{P}$, contradicting our assumptions. This completes the proof. □

Next we turn our attention to finding a $\mathcal{P}$-tight separation.

Find $\mathcal{P}$-tight
Input: A graph G, $T \subseteq V(G)$, $\mathcal{P}$-oracle f for a set $\mathcal{P}$ of connected subgraphs of $G - T$.
Find: A separation (X,Y) of order at most $|T|$ which is $\mathcal{P}$-tight for the pair (G,T) and of minimum order among all such tight separations.

Lemma 2.4. *There exists an algorithm solving* Find $\mathcal{P}$-tight *running in time $4^{|T|}n^{O(1)}$ utilizing $O(|T| \cdot |V(G)|^2 4^{|T|})$ calls of the $\mathcal{P}$-oracle.*

Proof. Let $G, T \subseteq V(G)$, and a $\mathcal{P}$-oracle f for a set of connected subgraphs $\mathcal{P}$ in G be given. Observe that for any $X \subseteq V(G)$, the function f is a $\mathcal{P}'$-oracle for the subset $\mathcal{P}' \subseteq \mathcal{P}$ of elements of $\mathcal{P}$ contained in the subgraph $G[X]$.

We first use the algorithm given in Lemma 2.3 to check if T is $\mathcal{P}$-free. If T is not free, let (X_1, Y_1) be the separation returned by the algorithm. If T is free, let (X_1, Y_1) be the trivial separation (T, G) with T treated as the graph with vertex set T and no edges. Let $\mathcal{P}_1 = \{P \in \mathcal{P} : P \subseteq Y_1\}$. Note that $X_1 \cap Y_1$ is $\mathcal{P}_1$-free in Y_1 by the guarantee that (X_1, Y_1) is a minimum order separation separating T from an element of $\mathcal{P}$.

We now define inductively define separations (X_i, Y_i) with the following properties.

1. (X_i, Y_i) is a separation of G of order $|X_1 \cap Y_1|$ with $V(X_{i-1}) \subsetneq V(X_i)$.
2. There exists $P \in \mathcal{P}$ with $P \subseteq Y_i$.

Given (X_i, Y_i), for $i = 1, \dots, k$, we now describe how to either construct (X_{k+1}, Y_{k+1}) or determine that (X_k, Y_k) satisfies the desired properties for the output.

First, consider the case when $Y_k - (X_k \cap Y_k)$ has multiple connected components. Let C be a component of $Y_k - (X_k \cap Y_k)$ such that C contains an element of $\mathcal{P}$. Then the separation $(X_{k+1}, Y_{k+1}) = (G-C, G[V(C)\cup V(X_i\cap Y_i)]-E(G[X_i\cap Y_i]))$ satisfies 1 and 2.

Assume now that $Y_k - (X_k \cap Y_k)$ has exactly one component. For every $x \in X_k \cap Y_k$, x is adjacent a vertex of $Y_k - X_k$ by the fact that no smaller order separation separates T from an element of $\mathcal{P}$. Arbitrarily fix a neighbor x' of x in $Y_k - X_k$. We apply the algorithm of Lemma 2.3 on the graph Y_k, subset of vertices $V(X_k \cap Y_k) \cup \{x'\}$, and the set $\mathcal{P}' = \{P \in \mathcal{P} : P \subseteq Y_k - (V(X_k \cap Y_k) \cup \{x'\})\}$ of connected subgraphs. Assume we get a separation (X', Y') certifying that $V(X_k \cap Y_k) \cup \{x'\}$ is not $\mathcal{P}'$-free in Y_k. It must hold that (X', Y') is of order exactly $|X_k \cap Y_k|$ and that there is an element $P \in \mathcal{P}$ such that $P \subseteq Y' - X'$. Thus, $(X_{k+1}, Y_{k+1}) = (X_k \cup X', Y')$ satisfies 1 and 2 above. We arbitrarily fix (X_{k+1}, Y_{k+1}) among all such possibilities and continue.

To define (X_{k+1}, Y_{k+1}) given (X_k, Y_k) takes at $O(4^{|T|}|T|^2(n+m)n \cdot m)$ time and at most $n \cdot |T| \cdot 4^{|T|}$ calls to to the $\mathcal{P}$-oracle. As $|V(X_{k+1})| > |V(X_k)|$, in time $O(4^{|T|}|T|^2(n+m)n^2 \cdot m)$ with at most $n^2 \cdot |T| \cdot 4^{|T|}$ calls to to the $\mathcal{P}$-oracle, we find (X_k, Y_k) such that $Y_k - (X_k \cap Y_k)$ has exactly one component and for all $x \in X_k \cap Y_k$, the set $V(X_k \cap Y_k) \cup \{x'\}$ is free in Y_k. Without loss of generality, we may assume that the edges of $G[V(X_k \cap Y_k)]$ are contained in X_k

To complete the proof, it suffices to show that (X_k, Y_k) is a tight separation. If not, there exists a separation (U, W) with $X_k \subsetneq U$ and an element $P \in \mathcal{P}$ with $V(P) \subseteq W - U$. Note that the order of (U, W) must be the same as (X_k, Y_k) and $V(X_k) \subsetneq V(U)$. We have that $U \cap W \neq X_k \cap Y_k$, lest $Y_k = (X_k \cap Y_k)$ have multiple components. Thus there exists a vertex $x \in X_k \cap Y_k$ which is contained in $U - W$. The separation (U, W) contradicts the fact that $V(X_k \cap Y_k) \cup \{x'\}$ is free in Y_k. This completes the proof of the lemma. $\square$

2.3 Proof of Theorem 2.1 Before proceeding with the proof, we will need several lemmas. Let us first recall Ramsey's theorem.

Lemma 2.5. (Ramsey's Theorem) *Let c, r, and n be positive integers with $n \geq c^{rc}$. Given a c-coloring of the edges of an n-clique, we can find in polynomial time a monochromatic r-clique in the coloring.*

As an easy corollary, we show that a large (not necessarily induced) matching implies the existence of either a large clique, or a large induced biclique, or a large induced matching.

Lemma 2.6. *Let H be a graph and $r \geq 1$ a positive integer. If H contains $M_{5^{10r}}$ as a subgraph, then H either contains M_r as an induced subgraph or H contains $K_{r,r}$ as a subgraph. Moreover, we can find the desired subgraph in polynomial time.*

Proof. By Lemma 2.5, every clique with $c := 5^{10r}$ vertices such that the edges are colored one of five colors contains a clique subgraph of size $2r$ where all the edges are the same color.

Let H contain M_c as a subgraph, and let $\{x_i, y_i\}$ for $1 \leq i \leq c$ form the edges of the matching. Consider the clique on c vertices, with the vertices labeled $1, \ldots, c$. We define a 5-coloring of the edges as follows. For an edge of the clique ij with $i < j$, we color the edge:

1. color 1 if no edge of H has one end in $\{x_i, y_i\}$ and one end in $\{x_j, y_j\}$,
2. color 2 if x_i is adjacent x_j,
3. color 3 if y_i is adjacent y_j and $x_i \nsim x_j$,
4. color 4 if x_i is adjacent y_j and $x_i \nsim x_j$, $y_i \nsim y_j$,
5. color 5 if y_i is adjacent x_j and $x_i \nsim x_j$, $y_i \nsim y_j$, $x_i \nsim y_j$

where all adjacencies are in the graph H. This defines a 5-coloring of the edges of the clique. By our choice of c, there exists a subset of vertices of size $2r$ inducing a monochromatic subclique, and we can identify it in polynomial time. Without loss of generality, we may assume that the vertices $1 \leq i \leq 2r$ of the clique induce such a monochromatic clique. If the subclique has color 1, then H contains an induced matching of size $2r$. If the monochromatic clique has color 2 or 3, then H contains a clique subgraph of size $2r$. Finally, if the subclique has color 4 (respectively, 5), then the vertices $\{x_1, \ldots, x_r\} \cup \{y_{r+1}, \ldots, y_{2r}\}$ (respectively, $\{y_1, \ldots, y_r\} \cup \{x_{r+1}, \ldots, x_{2r}\}$) induce a $K_{r,r}$ subgraph of H. $\square$

Let G be a graph. Let $T \subseteq V(G)$ be an independent set where $\deg_G(v) = 1$ for all $v \in T$, and let H be a graph with $V(H) = T$. We define the set of *truncated valid paths* to be the set

$$\mathcal{P} = \{P - T : P \text{ is a valid path.}\}$$

Note that by our assumptions on T, every element of $\mathcal{P}$ is a path. Note as well that given G, T, and H, for any set $X \subseteq V(G)$, we can test whether $G[X]$ contains an element of $\mathcal{P}$ in time $O(|V(G)| + |E(G)| + |E(H)|)$.

Lemma 2.7. *Let G be a graph. Let $T \subseteq V(G)$ be an independent set where $\deg_G(v) = 1$ for all $v \in T$, and let H be a graph with $V(H) = T$. Let $\mathcal{P}$ be the set of truncated valid paths. Let $T' \subseteq T$ be a subset such that*

i. $H[T']$ contains a perfect matching and
ii. T' is $\mathcal{P}$-free in $G - (T \setminus T')$.

There exists an algorithm which takes as input G, T, H, and T' and produces in output one of the following:

1. *a subset Z of at most $|T'|(|T'| + 3)$ vertices intersecting every valid path in G;*
2. *a separation (X, Y) of $G - T$ of order at most $|T'| + 2$ such that both X and Y contain an element of $\mathcal{P}$;*
3. *a subset $\bar{T}$ such that $T' \subseteq \bar{T} \subseteq T$, $|\bar{T}| = |T'| + 2$, $H[\bar{T}]$ contains a perfect matching, and $\bar{T}$ is $\mathcal{P}$-free in $G - (T \setminus \bar{T})$.*

The algorithm runs in time $4^{|T'|}((|V(G) - T| + |E(G)|)^{O(1)}$.

Proof. Let G, T, H, and T' be given. Let $|V(G) - T| = n$, $|E(G)| = m$, $|E(H)| = m'$, and $|T'| = t$. By our assumptions on T, $P - T$ is a (non-empty) path for every valid path P. Thus, any set of vertices intersecting every element of $\mathcal{P}$ also intersects every valid path.

Let G' denote the graph $G - (T \setminus T')$. We first find a separation (X, Y) of G' which is tight for the pair (G', T') of minimal order. By assumption, (X, Y) has order t. Lemma 2.4 allows us to do this in time $4^t(n+t)^{O(1)}$. Note, we are using here that we can test for elements of $\mathcal{P}$ in time $O(n+m+m')$. Without loss of generality, we may assume that $E(G[V(X \cap Y)])$ is contained in $E(X)$.

We can determine in time $O(n + m + m')$ if the separation $(X-T, Y-T)$ of $G'-T$ contains an element of $\mathcal{P}$ in $X - T$ as well as $Y - T$. If so, we return the separation $(X - T, Y - T)$ satisfying 2.

We determine in time $O(n + m + m')$ if $X \cap Y$ intersects every element of $\mathcal{P}$. If it does, we return $X \cap Y$ satisfying 1. Thus, we may assume that all elements of $\mathcal{P}$ intersect a vertex of $Y - X$ and at least one element of $\mathcal{P}$ is contained in $Y - X$.

Let the vertices of $X \cap Y$ be $\{x_1, x_2, \ldots, x_t\}$. Each vertex x_i has a neighbor in $Y - X$, lest $(X, Y - x_i)$ form a separation of order $t - 1$ violating our assumption that T' is free. Arbitrarily fix x_i' to be a neighbor of x_i in $Y - X$ for all $i = 1, \ldots, t$. Let $\mathcal{P}_i'$ be the set of elements of $\mathcal{P}$ contained in $Y - (V(X) \cup \{x_i'\})$. For each $i = 1, \ldots, t$, we find a $\mathcal{P}_i'$-tight separation (U_i, W_i) for the pair $(Y, V(X \cap Y) \cup x_i')$ of minimal order using the algorithm of Lemma 2.4. We can do this in time $4^t(n + t)^{O(1)}$.

Let $Z := \bigcup_1^t (U_i \cap W_i) \cup (X \cap Y) \cup T'$. Then $|Z| \leq t(t+1)+2t = t(t+3)$. If Z intersects every valid path, we return Z to satisfy 2. We can check this in time $O(n + m + m')$, and therefore proceed assuming that there exists an valid path $\bar{P}$ which is disjoint from Z. Fix such a path $\bar{P}$ for the remainder of the proof; let $P = \bar{P} - T$ and let $\bar{T} = T' \cup \{V(\bar{P}) \cap T\}$. Given that the endpoints of $\bar{P}$ are H-adjacent, it follows that $H[\bar{T}]$ contains a perfect matching. We test in time $4^t(n + m)^{O(1)}$ if $\bar{T}$ is $\mathcal{P}$-free in $G - (T \setminus \bar{T})$. If it is, we return $\bar{T}$ satisfying 3.

Assume, to reach a contradiction, that $\bar{T}$ is not $\mathcal{P}$-free in $G - (T \setminus \bar{T})$. In time $4^t(n + m)^{O(1)}$, we find a tight separation $(\bar{C}, \bar{D})$ for the pair $(G - (T \setminus \bar{T}), \bar{T})$ of minimum order. It follows from Observation 1 that $(\bar{C}, \bar{D})$ is a separation of order either t or $t + 1$ with $\bar{T} \subseteq V(\bar{C})$. We check in time $O(m + n + m')$ if $\bar{C}$ contains an element of $\mathcal{P}$. If it does, we return $(\bar{C} - T, \bar{D} - T)$ as a separation satisfying 2. Thus, we may assume that no element of $\mathcal{P}$ is contained in $\bar{C}$.

Let (C, D) be the separation $(\bar{C} - (\bar{T} \setminus T'), \bar{D} - (\bar{T} \setminus T'))$ of the graph G' after deleting the two vertices $\bar{T} \setminus T'$. By Observation 1, (C, D) is $\mathcal{P}$-tight for the pair (G', T'). Thus (C, D) as well has order either t or $t + 1$, implying that at most one of the two vertices $\bar{T} \setminus T'$ is contained in $\bar{C} \cap \bar{D}$. We conclude that since the path P is not contained in C that it intersects $C \cap D$ in at least one vertex, and if the intersection is exactly one vertex, then (C, D) has order strictly less than the order of $(\bar{C}, \bar{D})$, namely the order of (C, D) is equal to t.

We now show that (C, D) has order $t + 1$ and $X \subseteq C$. We check in time $O(n + m + m')$ whether $(C \cap D) \cup (X \cap Y)$ intersects every element of $\mathcal{P}$. If so, we return a set satisfying 1. Thus, we may assume that there exists an element $P' \in \mathcal{P}$ which is contained in $(D - C) \cap (Y - X)$. The separation $(X \cap C, Y \cup D)$ separates T' from the element P' of $\mathcal{P}$; thus it has order at least t. It follows that the order of the separation $(X \cup C, Y \cap D)$ must be of the same order as the separation (C, D). If $(X \cup C, Y \cap D)$ has order t, it follows that $(C, D) = (X, Y)$; but this is a contradiction as P intersects $C \cap D$ and was chosen to be disjoint from $X \cap Y$. We conclude that (C, D) has order $t + 1$. If $C \subsetneq X \cup C$, then $(X \cup C, Y \cap D)$ contradicts the tightness of (C, D). This proves the claim that (C, D) has order $t + 1$ and $X \subseteq C$.

The path P intersects $C \cap D$ in at least two vertices since some internal vertex of P must intersect $D - C$; it follows that there is at least one vertex of $X \cap Y$ which is contained in $C - D$, say x_i, and therefore x_i' is contained in C as well. Consider the separation (U_i, W_i). By Lemma 2.1, $U_i \cap C = X$. However, we have just showed that vertex x_i' is also an element of $U_i \cap C$.

This contradiction shows that $\bar{T}$ is $\mathcal{P}$-free in $G - (T \setminus \bar{T})$, completing the proof of correctness for the algorithm. The total runtime is $4^t(n + m)^{O(1)}$, as desired. ☐

We are now ready to proceed with the proof of Theorem 2.1.

Proof. (of Theorem 2.1.) Let G, T, H, k, and r be given. Let $n = |V(G) - T|$, $m = |E(G)|$, and $m' = |E(H)|$. Let $\mathcal{P}$ be the set of truncated valid paths.

Beginning with $T' = \emptyset$, we reiterate the algorithm from Lemma 2.7 up to $5^{10(k+r)}$ times to find one of the following:

1. a subset Z of at most $4 \cdot 5^{20(k+r)}$ vertices intersecting every path in G whose endpoints are H-adjacent;

2. a separation (X,Y) of $G-T$ of order at most $2\cdot 5^{10(k+r)}$ such that both X and Y contain an element of $\mathcal{P}$;
3. a subset T' such that $T'\subseteq T$, $|T'|=2\cdot 5^{10(k+r)}$, $H[T']$ contains a perfect matching, and T' is $\mathcal{P}$-free in $G-(T\setminus T')$.

At each iteration of the algorithm from Lemma 2.7, note that $|T'|\leq 2\cdot 5^{10(k+r)}-2$, so that if we ever find the hitting set Z in outcome 1 of Lemma 2.7, $|Z|\leq(2\cdot 5^{10(k+r)}-2)(2\cdot 5^{10(k+r)}+1)\leq 4\cdot 5^{20(k+r)}$, as desired.

Given the runtime of the algorithm from Lemma 2.7, we find one of the outcomes 1-3 above in time $4^{2\cdot 5^{10(k+r)}}(n+m)^{O(1)}\cdot 5^{10(k+r)}$. If we find the hitting set in outcome 1, we return Z and the algorithm terminates. Thus, we may assume we find either outcome 2 or 3.

Assume, as a case, we find a separation (X,Y) of $G-T$ satisfying outcome 2. We find valid paths P_X and P_Y such that P_X-T (resp. P_Y-T) is contained in X (resp. Y). This can be done in time $O(n+m+m')$. Define the graphs G_X with vertex set $V(X)\setminus V(Y)$ along with the vertices of $T-P_Y$ with a neighbor in $X-Y$ and edge set consisting of every edge with at least one end in $V(X)\setminus V(Y)$. We analogously define G_Y. Let $T_X=V(G_X)\cap T$ and similarly define T_Y. We can find G_X, G_Y, T_X, and T_Y in time $O(n+m)$. Similarly, we find the induced subgraphs H_X and H_Y with vertex sets $T\cap V(G_X)$ ($T\cap V(G_Y)$, respectively) in time $O(m')$. Let n_X and m_X (n_Y and m_Y) be $|V(G_X)\setminus T|$ and $|E(G_X)|$ (resp. $|V(G_Y)\setminus T|$ and $|E(G_Y)|$). Note that $n_X+n_Y\leq n$ and $m_X+m_Y\leq m$. We recursively run the algorithm on G_X, H_X, T_X, $k-1$ and on G_Y, H_Y, T_Y, $k-1$. The runtime of the recursive calls is $4^{3\cdot 5^{10(k+r)}}[(n_X+m_X)^{O(1)}+(n_Y+m_Y)^{O(1)}]$. If we find $k-1$ valid paths in X or Y, we return these paths along with either the path P_X or P_Y and the algorithm terminates. If we find sets Z_X and Z_Y hitting all the valid paths in the respective subgraphs, we return the set $Z_X\cup Z_Y\cup V(X\cap Y)$. Note that

$$\begin{aligned}|Z_X\cup Z_Y\cup V(X\cap Y)|&\\ \leq 4\cdot 5^{20(k-1+r)}+4\cdot 5^{20(k-1+r)}+2\cdot 5^{10(k+r)}&\\ \leq 4\cdot 5^{20(k+r)}&\end{aligned}$$

as desired. Finally, if we find a subset of T_X or T_Y inducing a matching of size r, we return the subset to satisfy outcome 3.

We conclude that we find the set T' satisfying outcome 3 in the repeated iterations of the algorithm of Lemma 2.7. By Lemma 2.6, in polynomial time we can either find a subset of T' inducing a matching of size t in H, or find subsets $B_1,B_2\subseteq T'$, $B_1\cap B_2=\emptyset$, and $|B_1|=|B_2|=k$ such that every vertex in B_1 is H-adjacent every vertex in B_2. If we find an induced matching of size t in H, we return that subgraph; thus we may assume we have subsets B_1 and B_2 as above.

We attempt to find k disjoint paths linking B_1 and B_2 in $G-(T\setminus T')$. If such paths exist, then we have found k disjoint valid paths as desired. Thus, we may assume there exists a separation of order at most $k-1$ separating the sets B_1 and B_2. Assume we include all the vertices of $T'\setminus V(B_1\cup B_2)$ in the separator, and we conclude that there exists a separation (X',Y') of order at most $|T'\setminus V(B_1\cup B_2)|+k-1$ with $B_1\subseteq V(X')$ and $B_2\subseteq V(Y')$ and $T'\setminus V(B_1\cup B_2)\subseteq X'\cap Y'$. Moreover, we can find the separation (X',Y') in polynomial time. We check in time $O(n+m+m')$ whether $X'-(B_1\cup(X'\cap Y'))$ and $Y'-(B_2\cup(X'\cap Y'))$ contain an element of $\mathcal{P}$. If not, we return $(X'\cap Y')\cup B_1\cup B_2$ as a set of at most $|T'|+k-1$ vertices intersecting all valid paths. Otherwise, without loss of generality, we assume $Y'-(B_1\cup(X'\cap Y'))$ contains an element of $\mathcal{P}$. The separation $(X'\cup B_2,Y')$ is a separation of $G-(T\setminus T')$ of order at most $|T'|-1$ with $T'\subseteq V(X'\cup B_2)$ separating T' from an element of $\mathcal{P}$, contrary to our assumptions on T'. □

3 Excluding induced matchings and skew bicliques: the exact FPT algorithm

The goal of this section is to prove the algorithmic part of Theorem 1.3: an FPT algorithm for Maximum Disjoint $\mathcal{H}$-Paths if $\mathcal{H}$ does not contain arbitrarily large induced matchings and skew bicliques. We state the algorithm in a robust way: even if the demand graph H contains large induced matchings and skew bicliques, the algorithm works, but either returns a correct answer or returns a large induced matching or a skew biclique of the demand graph H.

Theorem 3.1. *There is an algorithm that, given an instance (G,T,H,k) of* Maximum Disjoint Paths *and an integer r, in time $f(k,r)\cdot n^{O(1)}$ either*

- *finds k pairwise vertex-disjoint valid paths,*
- *correctly states that there is no set of k pairwise vertex-disjoint valid paths,*
- *returns an induced matching of size r in H, or*
- *returns an induced skew biclique of size $2r$ in H.*

We do not estimate the function $f(k,r)$ of Theorem 3.1 here, but as the algorithm eventually depends on the Disjoint Paths algorithm (Theorem 1.1), it is a tower of some number of exponentials.

Similarly to Section 2, by attaching a new degree-1 vertex to every terminal and moving the endpoints of the demand edges to these vertices, we may assume that T is an independent set of degree-1 vertices. As an opening step, we invoke the algorithm of Theorem 2.1 from Section 2. If it returns a solution with k pairwise disjoint valid paths, then we are done. Otherwise, the algorithm returns a hitting set Z of size $2^{O(k+r)}$ that covers every valid path, that is, for any connected component C of $G-Z$, no two vertices of $V(C)$ are adjacent in H. As every terminal is degree-1, we may assume that Z is disjoint from T: if some terminal t is in Z, then we may replace it with its unique neighbor. In this section, we assume that such a set Z is available and use the structural information given by Z to solve the problem.

If the number of terminals can be bounded by a function of k, then we can enumerate every sequence (s_1,t_1), ..., (s_k,t_k) of k pairs of terminals such that s_i and t_i are adjacent in H and invoke the algorithm of Theorem 1.1 for each such sequence. Therefore, our goal is to reduce number of terminals to a constant depending only on k. The main tool for this reduction is the notion of irrelevant terminals.

Given an instance (G,T,H,k) of Maximum Disjoint Paths, we say that a terminal $t \in T$ is *irrelevant* if (G,T,H,k) is a yes-instance if and only $(G,T-t,H,k)$. Note that, formally, if (G,T,H,k) is a no-instance, then every terminal is irrelevant. In a yes-instance, if there are more than $2k$ terminals, then some terminal is surely irrelevant. However, the main question is whether we can identify provably irrelevant terminals in a reasonable running time. The main technical result of the section is showing that if we have a bounded-size hitting set Z of the valid paths and there are many terminals, then we can identify an irrelevant terminal in FPT time. Therefore, we can remove that vertex from the set of terminals and repeat the process until the number of terminals becomes bounded by a constant depending only on k. We formulate the following result in such a way that the algorithm either finds an irrelevant terminal, a solution with k disjoint paths, or one of the forbidden induced subgraphs in H (induced matchings or skew bicliques).

Lemma 3.1. *For every k, r and z, there is a constant $I_{k,r,z}$ such that the following holds. Let (G,T,H,k) be an instance of* Maximum Disjoint Paths, *let r be an integer, and let $Z \subseteq V(G) \setminus T$ be a set of at most z vertices such that $G-Z$ does not contain a valid path. If $|T| > I_{k,r,z}$, then in time $f(k,r,z) \cdot n^{O(1)}$, we can either*

- *find an irrelevant terminal $x \in T$,*
- *a set of k pairwise disjoint valid paths,*
- *return an induced matching of size r in H, or*
- *return an induced skew biclique of size $2r$ in H.*

Theorem 3.1 follows easily from Lemma 3.1 and Theorem 1.1.

Proof. (of Theorem 3.1) Let us modify first the instance such that T is an independent set of degree-1 vertices in G. Let us invoke the algorithm of Theorem 2.1. If it returns a solution with k vertices or an induced matching of size r in H, then we are done. Otherwise, we get a set Z of vertices that covers every valid path. We may assume that Z is disjoint from T, as we can replace any terminal in Z with its unique neighbor: the resulting set still has the property that covers every valid paths.

If $|T| > I_{k,r,z}$, then we invoke the algorithm of Lemma 3.1. If it returns an induced matching or a skew biclique in H, then we are done. Otherwise, if it returns an irrelevant terminal v, then we remove v from the set of terminals, that is, we continue with the instance $(G, T \setminus \{v\}, H-v, k)$. We repeat this steps as long as $|T| > I_{k,r,z}$ holds. If $|T| \leq I_{k,r,z}$, then we enumerate every sequence (s_1,t_1), ..., (s_k,t_k) of pairs of vertices from T such that s_i and t_i are adjacent in H. There are at most $|T|^{2k} \leq I^{2k}_{k,r,z}$ such sequences, which is number that can be bounded by a function of k, z, and r only. For each such sequence, we use Theorem 1.1 to find disjoint paths connecting these pairs of vertices. It is clear that the Maximum Disjoint Paths instance has a solution if and only if the algorithm of Theorem 1.1 returns k disjoint paths for at least one of these sequences. □

The proof of Lemma 3.1 appears in Sections 3.1–3.4. Let us review here the main ideas of the proof.

Handling large cliques in H. As a first step, we show how to find an irrelevant terminal given a large clique K of H (Section 3.1). The special case of the disjoint paths problem when the demand pattern is a clique is a well-understood problem and we can use standard polynomial-time algorithms to find k disjoint paths with endpoints in K. If there are k such paths, then they form a solution of the instance. Otherwise, a classical result of Gallai [8] shows that there is a small set S of vertices that cover every path with both endpoints in K, or in other words, every connected component of $G-S$ contains at most one vertex of K. Then we use this information to identify a vertex of K that is an irrelevant terminal.

Lemma 3.2. *There is polynomial-time algorithm that, given an instance of* Maximum Disjoint Paths *and a clique K of H having size $10k^2$, either*

- *returns a set of k pairwise disjoint valid paths, or*

- *returns an irrelevant terminal t.*

Separations and representative sets. Given a component C of $G - Z$, we can define a separation (A, B) with $C = V(A) \setminus V(B)$, which has the property that no two vertices of A are adjacent in H. The main technical part of the proof is showing that if A contains many terminals in such a separation, then we can find an irrelevant vertex.

LEMMA 3.3. *For every k, r and z, there is a constant $I^{\text{sep}}_{k,r,z}$ such that the following holds. Let (A, B) be a separation of order at most z with $|T \cap V(A)| > I^{\text{sep}}_{k,r,z}$, $V(A) \cap V(B)$ disjoint from T, and H has no edge in $V(A)$. In time $f(k, r, z) \cdot n^{O(1)}$, we can either*

- *find an irrelevant terminal $x \in T \cap V(A)$,*
- *return a set of k pairwise disjoint valid paths,*
- *return an induced matching of size r in H, or*
- *return an induced skew biclique of size $2r$ in H.*

Given a solution and a separation (A, B), let us focus on the part of the solution inside A. An obvious and standard way of approaching the problem would be to define an equivalence relation on these partial solutions, where two partial solutions are equivalent if any way of extending one of them to a full solution with edges in B is also a valid extension of the other partial solution. Let us enumerate one partial solution from each equivalence class. If a terminal t in A is not used by any of the enumerated partial solutions, then it is irrelevant: if a partial solution is using t, then there is an equivalent partial solution not using t, hence the solution can be modified not to use t. If the number of equivalence classes is bounded by a constant, then this gives a way of finding an irrelevant terminal if the number of terminals in A is larger than a constant.

Unfortunately, in our problem, the number of equivalence classes cannot be bounded by any function of k and the size of the separation. A partial solution contains paths connecting a subset T' of terminals in A to the separator $V(A) \cap V(B)$, and the equivalence class of the partial solution depends on what exactly this subset is, as it determines which terminals can complete these paths to valid paths of the solution. Therefore, the number of different types a partial solution can have cannot be bounded by a function of k and the order z of the separation only: it depends also on the number $|T|$ of terminals and can be as large as $\Omega(|T|^z)$. For example, let G be a star with center v and $2n$ leaves a_1, $\dots$, a_n, b_1, $\dots$, b_n. Let $A = G[\{v, a_1, \dots, a_n\}]$ and $B = G[\{v, b_1, \dots, b_n\}]$. Now (A, B) is a separation of order 1. Let $T = V(G) \setminus \{v\}$ and let H be the matching with edges a_1b_1, $\dots$, a_nb_n. Let $k = 1$. Now for $1 \le i \le n$, the partial solutions consisting of the single edge a_iv are in different equivalence classes: the edge vb_i extends a_iv to a solution, but it does not extend a_jv for any $j \neq i$. Therefore, there are n equivalence classes of partial solutions. By taking disjoint unions of such stars, the reader may modify this example for larger $k = z$ such that the number of terminals is $2kn$ and the number equivalence classes is n^k.

One may hope that by excluding large induced matchings and large skew bicliques, the number of equivalence classes can be bounded by a constant. Let us point out by a simple example that this is not the case. Let us modify the example in the previous paragraph such that H is now a complete bipartite graph minus the edges a_ib_i for $1 \le i \le n$. Observe that H does not contain large induced matchings and large induced skew bicliques. Again, for $1 \le i \le n$, the partial solutions consisting of the single edge a_iv are in different equivalence classes: the edge vb_i extends a_jv to a solution for any $j \neq i$, but it *does not* extend a_iv. Therefore, again we have n equivalence classes of partial solutions.

We get around this problem using the idea of representative sets. We show that, even though the valid partial solutions in a small separation may form an unbounded number of equivalence classes, they have a bounded-size subset that is representative in the sense that if any partial solution can be extended to a correct solution, then one of the partial solutions in the representative set can also be extended to a correct solution. Continuing our example from the previous paragraph, even though there are n incomparable partial solutions, there is a representative set consisting of only two partial solutions, the edge a_1v and the edge a_2v. Indeed, if a solution contains the edge vb_1, then the part of the solution in A can be replaced with a_2v; if a solution contains the edge vb_i for $1 < i \le n$, then the part of the solution in B can be replaced by a_1v. We show how to find a representative set of partial solutions of bounded size. Then any terminal in A that is not used by any of these partial solutions can be considered to be irrelevant. The bound and the algorithm relies heavily on the assumption that the graph H does not contain large cliques, large induced matchings, skew bicliques graphs; or more precisely, the algorithm either works correctly, or returns one such graph. If we find a clique, then we can invoke Lemma 3.2. By the specification of Lemma 3.1, the induced matchings or skew biclique can be returned. The concept of representative sets has been used in the design of FPT algorithms [21, 6, 7, 18, 16], but our application does not follow from any of the earlier technical statements; in particular, the fact that this approach works precisely when there are no larges cliques, bicliques, or matchings is quite specific to our problem.

On a high level, the proof of Lemma 3.3 goes the following way. Consider those paths of the solution that cross the separator and have one endpoint in A and one in B. The endpoints in A form a vector $\mathbf{a}$ and the endpoints in B form a vector $\mathbf{b}$. These two vectors are compatible in the sense that the j-th coordinate of $\mathbf{a}$ is adjacent in H with the j-th coordinate of $\mathbf{b}$. The partial solution connects the vertices in $\mathbf{a}$ to the separator $V(A) \cap V(B)$. If we want to replace the partial solution with another partial solution that connects a different set $\mathbf{a}'$ of vertices to the separator, then we have to make sure that the new vector $\mathbf{a}'$ is also compatible with the vector $\mathbf{b}$. Therefore, if we classify the partial solution according to the vector of terminals connected to the separator, then we have to find a representative subset of these vectors in the sense that if some vector $\mathbf{a}$ is compatible with some vector $\mathbf{b}$, then the representative subset also contains a vector $\mathbf{a}'$ compatible with $\mathbf{b}$. In Section 3.2, we consider this abstract problem on vectors, and show (assuming that H has no large induced matching or induced skew biclique) how we can find a bounded-size representative set of vectors. Of course, our problem is more complicated than just matching these vectors, for example, a path in the solution can cross the separator several times. In Section 3.3, we address these issues by classifying the partial solutions into a bounded number of types according (mostly) to what happens at the separator. We conclude the proof of Lemma 3.3 in Section 3.3.

Reducing the number of components. Finally, after we reduced the number of terminals in each component of $G - Z$ with repeated applications of Lemma 3.3, our goal is to reduce the number of components of $G - Z$ that contain terminals. In Section 3.4, we show that this can be done quite easily by a simple marking procedure. The proof relies on the fact that the number of terminals is bounded in each component. Thus it does not seem to be easy to do the reduction of the number of components before the reduction of the number terminals in the components.

3.1 Handling cliques

In this section, we discuss how to find an irrelevant vertex if we have a large clique in the demand graph H (Lemma 3.2 above). The reason why we are treating this special case separately is that a combinatorial argument of the following section (Lemma 3.4) works only if we can assume that there are no large induced matchings, skew bicliques, and cliques in the demand graph H. By the specification of Theorem 3.1, if we encounter large induced matchings or skew bicliques, then we may stop, but there is no reason why large cliques cannot appear in the demand graph H. Therefore, we need some argument to handle large cliques, and this is what we provide in this section. Note that even if we have a procedure handling large cliques, we cannot say the we apply it exhaustively on every sufficiently large clique of H and after that it can be assumed that H has no large cliques: finding a clique of size k is W[1]-hard. Instead, what we do is whenever the algorithm described in the following section fails because it finds a a large clique, then we invoke this procedure.

The following result was proved by Gallai [8] in a combinatorial form, the algorithmic version is folklore:

Theorem 3.2. (Gallai [8]) *Given an undirected graph G, a set $A \subseteq V(G)$ of vertices, and an integer k, we can find in polynomial time either*

- *a set of k pairwise vertex-disjoint paths with endpoints in A, or*
- *a set S of at most $2k-2$ vertices such that every component of $G - S$ contains at most one vertex of $A \setminus S$.*

Using Theorem 3.2 on a sufficiently large clique K of H, we may either find k valid paths forming a solution or we can identify a terminal of K that can be always avoided in a solution.

Proof. (of Lemma 3.2) By Theorem 3.2 applied to graph G, vertices K, and integer k, we can find in polynomial time either k pairwise vertex-disjoint paths with endpoints in K, or a set S of size at most $2k-2$ such that every component of $G-S$ contains at most one vertex of K. In the former case, we return this set of k paths as a valid solution. In the later case, let $S' \subseteq S$ contain a vertex $v \in S$ if there are at least $5k+1$ components of $G - S$ that are adjacent to v and intersect K (in exactly one terminal). This means that there are at most $5k|S \setminus S'| \le 5k|S| \le 5k \cdot (2k-2) = 10(k^2 - k)$ components C of $G - S$ with $C \cap K \neq \emptyset$ and $N(C) \not\subseteq S'$. As $|K| \ge 10k^2$ and hence $|K \setminus S| \ge 10k^2 - (2k-2) > 10(k^2-k)+k$, there exists k components $C_1, \ldots, C_k$ with $|C_i \cap K| = 1$ and $N(C_i) \subseteq S'$. If each of these k components fully contains a valid path, then picking a valid path from each of them gives a solution that we can return. Otherwise, there is a component C of $G - S$ with $C \cap K = \{t\}$, $N(S) \subseteq S'$, and not containing a valid path.

We claim that removing t from the set of terminals gives an equivalent instance. That is, we show that any solution containing a path P with endpoint t can be modified in such a way that it does not use t.

By the choice of C, there is no valid path in C, hence we know that P is not contained fully in C. Let v be the vertex of $N(C) \subseteq S'$ that is closest to t on P. As $v \in S'$, there are at least $5k+1$ components of $G - S'$ intersecting K and adjacent to v. At most $k-1$ of them can contain fully a path of the solution (different from P) and at most $2|S| \leq 4k$ of them can contain a path going intersecting $|S|$ (observe that a path containing x vertices of $|S|$ can intersect at most $x + 1 \leq 2x$ components). Therefore, there are two such a components C^1, C^2 disjoint from every path of the solution; let $C^1 \cap K = \{t_1\}$ and $C^2 \cap K = \{t_2\}$. Now the path P can be replaced by a path connecting t_1 and t_2 via v. This proves the claim that removing t from the set of terminals gives an equivalent instance. $\square$

3.2 Representative sets for vectors of vertices

In this section, we prove a statement about representative sets in an abstract setting of compatible vectors (Lemma 3.6 below). In Section 3.3, we use this result to prove a bound on the size of representative sets of partial solutions, which will allow us to find irrelevant terminals if a component of $G - Z$ contains too many terminals.

Definition 4. *Let H be an undirected graph and let d be a positive integer. We say that two d-tuples $(a_1, \dots, a_d), (b_1, \dots, b_d) \in V(H)^d$ are* compatible *if a_i and b_i are adjacent in H for every $1 \leq i \leq d$. Let $\mathcal{R} \subseteq V(H)^d$ be a set of d-tuples. We say that $\mathcal{R}' \subseteq \mathcal{R}$ is a* representative *subset of $\mathcal{R}$ if for every compatible pair $\mathbf{a} \in \mathcal{R}$ and $\mathbf{b} \in V(H)^d$, there is an $\mathbf{a}' \in \mathcal{R}'$ such that $\mathbf{a}'$ and $\mathbf{b}$ are compatible.*

Note that we do not require that the coordinates of a vector $(a_1, \dots, a_d)$ be all distinct, and $(a_1, \dots, a_d)$ and $(b_1, \dots, b_d)$ can be compatible even if $a_i = b_j$ for some $i \neq j$ (but $a_i = b_i$ is clearly impossible, as no vertex of H is adjacent to itself).

We need the following simple Ramsey argument, whose proof is very similar to the proof of Lemma 2.6 in Section 2.

Lemma 3.4. *Let r and n be positive integers with $n \geq 4^{4r}$. Let H be a graph and $a_1, \dots, a_n, b_1, \dots, b_n$ be distinct vertices such that*

- *a_i and b_i are adjacent for $1 \leq i \leq n$, and*
- *a_i and b_j are not adjacent for $1 \leq i < j \leq n$.*

Then in polynomial time we can find either

- *an induced matching of size r in H,*
- *an induced skew biclique on $2r$ vertices in H, or*
- *a clique of size r in H.*

The following lemma states that (assuming there is no large induced matching, skew biclique, or clique in H) every set of vectors has a bounded-size representative subset.

Lemma 3.5. *Let H be an undirected graph, r and d positive integers, and $\mathcal{R} \subseteq V(H)^d$ a set of d-tuples. Suppose that there is no induced matching of size r, induced skew biclique of size $r + r$, or clique of size r in H. Then there is a representative subset $\mathcal{R}' \subseteq \mathcal{R}$ of size at most $R^{\text{vec}}_{d,r} := (d+1)^{d(4^{4r})}$.*

We prove an algorithmic version of Lemma 3.5. The straightforward algorithmic statement would be to say that, given a set $\mathcal{R}$ of vectors, a bounded-size representative set can be found. However, we would like to find small representative sets efficiently also for large, implicitly given sets $\mathcal{R}$ that would be too time consuming to enumerate explicitly. Therefore, we state the algorithmic version of Lemma 3.5 in a way that $\mathcal{R}$ is given by a query procedure that, given sets $A_1, \dots, A_d \subseteq V(H)$, returns a vector $\mathbf{a} \in (A_1 \times \dots \times A_d) \cap \mathcal{R}$, if such a vector exists.

Lemma 3.6. *Let H be an undirected graph, r and d positive integers, and $\mathcal{R} \subseteq V(H)^d$ a set of d-tuples. Suppose that the set $\mathcal{R}$ is given via a query procedure that, given sets $A_1, \dots, A_d \subseteq V(H)$, returns an $\mathbf{a} \in (A_1 \times \dots \times A_d) \cap \mathcal{R}$, or states that no such vector $\mathbf{a}$ exists. There is an algorithm whose running time is polynomial in n, in $R^{\text{vec}}_{d,r} := (d+1)^{d(4^{4r})}$, and in the running time of the query procedure, and finds either*

- *a representative subset $\mathcal{R}' \subseteq \mathcal{R}$ of size at most $R^{\text{vec}}_{d,r}$,*
- *an induced matching of size r in H,*
- *an induced skew biclique on $2r$ vertices in H, or*
- *a clique of size r in H.*

Proof. The algorithm builds a rooted tree where each node is either empty or contains a compatible pair $(\mathbf{a}, \mathbf{b})$ with $\mathbf{a} \in \mathcal{R}$ and $\mathbf{b} \in V(H)^d$. Empty nodes have no children and each nonempty node has exactly d ordered children. Initially, we start with a tree consisting of a single empty node.

For a vector $\mathbf{b}' \in V(H)^d$, we define the following search procedure on the tree. We start the procedure at the root node. If the current node is empty, then we say that the procedure fails at this empty node. Otherwise, let $(\mathbf{a}, \mathbf{b})$ be the current node. If $\mathbf{a}$ and $\mathbf{b}'$ are compatible, then we declare the search to be successful. Otherwise, let $1 \leq j \leq d$ be the first coordinate such that the j-th coordinates of $\mathbf{a}$ and $\mathbf{b}'$ are not adjacent. Then we continue the search at the j-th child of the current node.

Given an empty node u of the tree, we show how to check whether there are d-tuples $\mathbf{a}' = (a'_1, \ldots, a'_d) \in \mathcal{R}$ and $\mathbf{b}' = (b'_1, \ldots, b'_d) \in V(H)^d$ such that $\mathbf{a}'$ and $\mathbf{b}'$ are compatible and $\mathbf{b}'$ fails at u. Consider the path from the root of the tree to the empty node u. Let $(\mathbf{a}, \mathbf{b})$ be a nonempty node on this path such that the path continues with the j-th child of this nonempty node. Then the j-th coordinate of $\mathbf{a}$ is *not* adjacent to the j-th coordinate of $\mathbf{b}'$, while for every $1 \leq j' < j$, the j'-th coordinate of $\mathbf{a}$ is adjacent to the j'-th coordinate of $\mathbf{b}'$. These requirements together give a subset $B_j \subseteq V(H)$ of potential values for the j-th coordinate of $\mathbf{b}'$. Now a vector $\mathbf{b}' \in V(H)^d$ fails at u if and only if $\mathbf{b}' \in B_1 \times \cdots \times B_d$. Therefore, we need to find a vector $\mathbf{a}' \in \mathcal{R}$ that is compatible with at least one vector in $B_1 \times \cdots \times B_d$. Let A_j contain every vertex of H that has at least one neighbor in B_j. Observe that a vector $\mathbf{a}' \in V(H)^d$ is compatible with at least one vector in $B_1 \times \cdots \times B_d$ if and only if $\mathbf{a}' \in A_1 \times \cdots \times A_d$. Therefore, we can use the query procedure to check the existence of such a vector $\mathbf{a}' = (a'_1, \ldots, a'_d)$ and then we can construct $\mathbf{b}' = (b'_1, \ldots, b'_d) \in B_1 \times \cdots \times B_d$ by letting b'_j be an arbitrary neighbor of a'_j in B_j.

We consider every empty node u (in arbitrary order) and use the method described in the previous paragraph to find an $\mathbf{a}' \in \mathcal{R}$ and a d-tuple $\mathbf{b}'$ compatible with $\mathbf{a}'$ that fails at u. If there is such a pair $(\mathbf{a}', \mathbf{b}')$, then we replace u with $(\mathbf{a}', \mathbf{b}')$ and add d empty children to this node. We repeat this step until no such $\mathbf{b}'$ can be found for any empty node u. At this point, let us define the set $\mathcal{R}' = \{\mathbf{a} \mid (\mathbf{a}, \mathbf{b}) \text{ appears in a nonempty node}\}$, that is, $\mathcal{R}'$ contains the first part of every pair appearing in the tree. Clearly, we have $\mathcal{R}' \subseteq \mathcal{R}$ from the way new nonempty nodes are introduced into the tree. Moreover, we claim that $\mathcal{R}'$ is a representative subset of $\mathcal{R}$. Indeed, for every adjacent pair $\mathbf{a}' \in \mathcal{R}$ and $\mathbf{b}' \in V(H)^d$, the search procedure for $\mathbf{b}'$ cannot fail at any empty node u (otherwise we would have extended the tree at u) and therefore the tree contains a pair $(\mathbf{a}, \mathbf{b})$ such that $\mathbf{a} \in \mathcal{R}'$ is compatible with $\mathbf{b}'$.

We prove that if the height h of the tree reaches $d(4^{4r})$, then we can find an induced matching, a skew biclique, or a clique of the specified size and we can stop the algorithm. Otherwise, if the algorithm terminates without stopping this way, then every path from the root to a leaf contains less than $d(4^{4r})$ nonempty nodes, and hence the number of nonempty nodes is at most $\sum_{i=0}^{d(4^{4r})-1} d^i \leq (d+1)^{d(4^{4r})} = R^{\text{vec}}_{d,r}$. Therefore, as showed in the previous paragraph, we obtain a representative subset $\mathcal{R}'$ of size at most $R^{\text{vec}}_{d,r}$.

Consider a path from the root to a leaf with at least $d(4^{4r})$ nonempty nodes. Then there is a $1 \leq j \leq d$ such that it is true for at least $n = 4^{4r}$ nodes on the path that the path continues with the j-th child of the node. Let $(\mathbf{a}_1, \mathbf{b}_1)$, $\ldots$, $(\mathbf{a}_n, \mathbf{b}_n)$ be n such nodes, ordered as they appear on the path from the root to the leaf. Let a_i and b_i be the j-th coordinate of $\mathbf{a}_i$ and $\mathbf{b}_i$, respectively. As the pair $(\mathbf{a}_i, \mathbf{b}_i)$ is compatible, we have that a_i and b_i are adjacent. Furthermore, consider the execution of the search procedure when $\mathbf{b}_i$ failed and the node $(\mathbf{a}_i, \mathbf{b}_i)$ was added to the tree. Note that the tree is extended only by replacing leaf nodes, thus the ancestors of $(\mathbf{a}_i, \mathbf{b}_i)$ did not change after they were added to the tree. Therefore, for every $1 \leq i' < i$, the search procedure for $\mathbf{b}_i$ encountered the node $(\mathbf{a}_{i'}, \mathbf{b}_{i'})$ and then continued the search with the j-th child of this node. This means that the j-th coordinate of $\mathbf{b}_i$ is not adjacent to the j-th coordinate of $\mathbf{a}_{i'}$. That is, we get that $a_{i'}$ is not adjacent to b_i for every $1 \leq i' < i \leq n$. Therefore, the conditions of Lemma 3.4 hold, and we can use it to return an induced matching, a skew biclique, or a clique. □

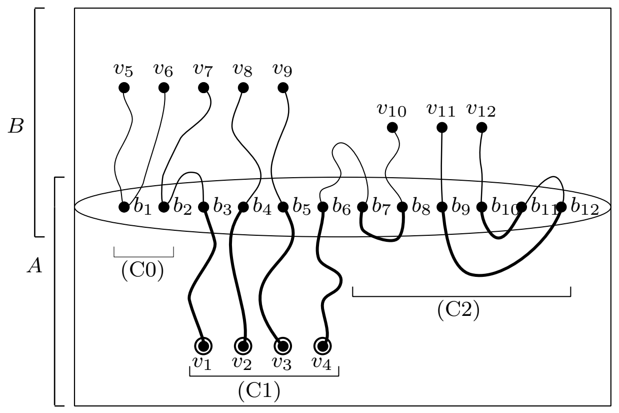
3.3 Representative sets for disjoint paths We can describe a solution as a subgraph P of G that is the union of k pairwise-disjoint valid paths. A *partial solution* is any subgraph of G that is the union of disjoint paths (possibly more than k or possibly with endpoints not in T). Given a solution P and a separation (A, B) of G, the *partial solution* of P at (A, B) is the subgraph Π of P induced by $V(A)$. To define representative sets of partial solutions, we need to define first what it means to replace a partial solution with another:

Definition 5. *Let P be a solution, let (A, B) be a separation of G, and let Π be a partial solution at (A, B). We say that Π is* replacable *at (A, B) in P if the subgraph $P' = (P - E(G[V(A)])) \cup \Pi$ is a valid solution. In this case, we say that P' is obtained by* replacing *Π into P at (A, B).*

Definition 6. *Let $\mathcal{R}$ be a set of partial solutions at (A, B). We say that $\mathcal{R}$ is* representative *if for every solution P, there is a $\Pi \in \mathcal{R}$ that is replacable into P at (A, B). We say that a subset $\mathcal{R}' \subseteq \mathcal{R}$* represents *$\mathcal{R}$ if for every solution P whose partial solution at (A, B) is in $\mathcal{R}$, there is a $\Pi \in \mathcal{R}'$ that is replacable into P at (A, B).*

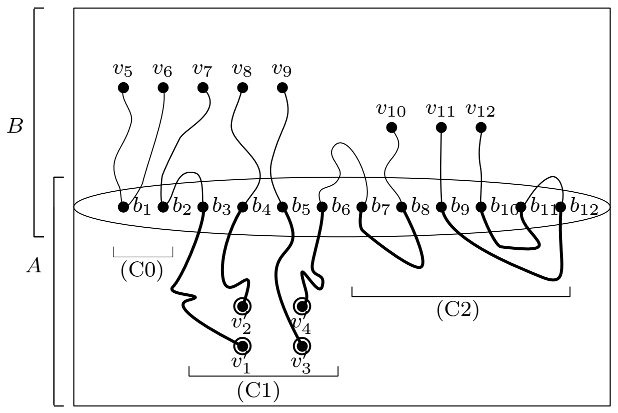
The main result of the section is the following:

Lemma 3.7. *For every k and z, there is a constant $R_{z,r}$ such that the following holds. Let (A, B) be a separation of order z such that $V(A) \cap V(B)$ is disjoint from T and H has no edge in $V(A)$. Let $\mathcal{R}$ contain*



B
A
v_5 v_6 v_7 v_8 v_9
v_{10} v_{11} v_{12}
b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8 b_9 b_{10} b_{11} b_{12}
(C0)
(C2)
v_1 v_2 v_3 v_4
(C1)

Figure 1: A partial solution at (A, B) (Lemma 3.7). Set $S_0 = \{b_1, b_2\}$ contains the vertices of the two paths of type (C0). There are four paths of class (C1), connecting $\{v_1, v_2, v_3, v_4\}$ to $J = \{b_3, b_4, b_5, b_6\}$. The three paths of type (C2) define the matching $M = \{b_7b_8, b_9b_{12}, b_{10}b_{11}\}$. Assuming the ordering (b_3, b_4, b_4, b_6) of J, the inner vector is (v_1, v_2, v_3, v_4) and the outer vector is (v_7, v_8, v_9, v_{10}).



B
A
v_5 v_6 v_7 v_8 v_9
v_{10} v_{11} v_{12}
b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8 b_9 b_{10} b_{11} b_{12}
(C0)
(C2)
v'_2 v'_4
v'_1 v'_3
(C1)

Figure 2: A partial solution having the same type as the partial solution in Figure 1, and replacing it at (A, B). The inner vector is (v'_1, v'_2, v'_3, v'_4). Assuming H contains the edges v'_1v_7, v'_2v_8, v'_3v_9, v'_4v_{10}, the result of the replacement is a valid solution.

the partial solution at (A, B) for every solution. In time $f(r, z) \cdot n^{O(1)}$, we can either

- *find a representative set $\mathcal{R}' \subseteq \mathcal{R}$ of partial solutions at (A, B) with $|\mathcal{R}'| \leq R_{z,r}$,*
- *return an induced matching of size r in H,*
- *return an induced skew biclique on $2r$ in H, or*
- *return a clique of r in H.*

Proof. Let $S = V(A) \cap V(B)$. Let P be a solution and let Π be the partial solution of P at (A, B). As H has no edge in $V(A)$, every path of the partial solution contains a vertex of S, hence there are at most z paths in the partial solution. Each path P can be classified into exactly one of the following three classes (recall that $S \cap T = \emptyset$); see Figure 1:

(C0) P consists of single vertex of S.
(C1) P has length at least one and has one endpoint in T and one endpoint in S.
(C2) P has length at least one and has both endpoints in S.

The paths of class (C2) define a (not necessarily perfect) matching M of S the obvious way. We define the *join vertex* of a path P of class (C1) to be its endpoint in S. Let $J \subseteq S$ be the join vertices of the paths of class (C1). We define the *type* of a partial solution Π_A to be the triple $\tau = (S_0, J, M)$, where

- $S_0 \subseteq S$ is the set of vertices used by paths of class (C0).
- $J \subseteq S$ is the set of join vertices of paths of class (C1),
- M is the matching of S defined above based on the paths of class (C2).

Note that the number of types is at most $T := 2^z \cdot 2^z \cdot z^z$. Let $\mathcal{R}_\tau \subseteq \mathcal{R}$ contain every partial solution of type τ. For every type τ, we construct a representative subset $\mathcal{R}'_\tau \subseteq \mathcal{R}_\tau$. It is clear that the union $\mathcal{R}'$ of $\mathcal{R}'_\tau$ for every type τ is representative subset of $\mathcal{R}$.

We construct $\mathcal{R}'_\tau$ for a type $\tau = (S_0, J, M)$ the following way. Let us fix an ordering of $J = (v_1, \dots, v_d)$ (note that $d \leq z$). For a partial solution Π of type τ, let P_j be the path of class (C1) whose join vertex is v_j. Let a_j be the other endpoint of P_j. We define the d-tuple $\mathbf{a} = (a_1, \dots, a_d) \in V(H)^d$ as the *inner vector* of the partial solution Π.

Let $\overline{\mathcal{R}}_\tau$ be the inner vectors of the partial solutions in $\mathcal{R}_\tau$. We would like to invoke Lemma 3.6 on the set $\overline{\mathcal{R}}_\tau$. For this purpose, we need to implement the query procedure. We need to test the existence of a partial solution of type τ whose inner vector is in $A_1 \times \cdots \times A_d$. We reduce this question to solving an instance of the k-disjoint paths problem. As we have observed earlier, each partial solution of type τ consist of a set of at most z vertex-disjoint paths. Let us start with the graph $A - S_0$: we remove the set S_0, as it is reserved for paths of class (C0). For every pair (v_1, v_2) in the matching M, we introduce a corresponding pair in the constructed Disjoint Paths instance: the paths of the solution connecting these pairs will correspond to the requested paths of class (C2) in the partial solution. To handle paths of type (C1), let us introduce a vertex s_j adjacent to every vertex of A_j for every $1 \leq j \leq d$. Then we specify the pairs (s_j, v_j) for every $1 \leq j \leq d$ (recall that the v_j's are the vertices of J). Let us use the algorithm of Theorem 1.1 to find vertex-disjoint paths with the specified endpoints. If such a collection of disjoint paths exist, then we obtain, after removing the vertices $s_1, \ldots, s_d$, a set of disjoint paths in A. These paths form a partial solution of type τ whose inner vector is in $A_1 \times \cdots \times A_d$, hence the query procedure can return this partial solution. Conversely, if there exists a partial solution Π of type τ having inner vector in $A_1 \times \cdots \times A_d$, then it gives a solution for the constructed instance of Disjoint Paths. This implies that the algorithm of Theorem 1.1 finds a solution for this instance of Disjoint Paths, resulting in a partial solution Π' of type τ and inner vector in $A_1 \times \cdots \times A_d$.

Using the query procedure described in the previous paragraph, we may invoke Lemma 3.6 on the set $\overline{\mathcal{R}}_\tau$. If we get an induced matching, induced skew biclique, or a clique, then we are done. Otherwise, we get a representative subset $\overline{\mathcal{R}}'_\tau$ of $\overline{\mathcal{R}}_\tau$ having size at most $R^{\text{vec}}_{d,r}$. Note that each vector $\mathbf{a}$ introduced into $\overline{\mathcal{R}}'_\tau$ was returned by the query procedure, which means that the query procedure found a partial solution of type τ and inner vector $\mathbf{a}$; let $\mathcal{R}'_\tau \subseteq \mathcal{R}_\tau$ contain every such partial solution. Finally, we construct the set $\mathcal{R}'$ as the union of $\mathcal{R}'_\tau$ for every type τ; as both the number of types and the size of each $\mathcal{R}'_\tau$ can be bounded by a function of r and z only, the size of $\mathcal{R}'$ can be bounded by a constant $R_{z,r}$ depending only on r and z.

We claim that $\mathcal{R}'$ is also a representative set of partial solutions at (A, B). Let P be a solution and let $\Pi \in \mathcal{R}$ be its partial solution at (A, B). Suppose that Π has type $\tau = (S_0, M, J)$ and let $\mathbf{a}$ be the inner vector of Π. Recall that we fixed an ordering $(v_1, \ldots, v_j)$ of J, there is a path P_j of type (C1) with endpoints a_j and v_j for every $1 \leq j \leq d$, and the inner vector is $(a_1, \ldots, a_j)$. For every $1 \leq j \leq d$, let b_j be the other endpoint of the path of a_j in the solution P. We define $\mathbf{b} = (b_1, \ldots, b_d) \in V(H)^d$ as the *outer vector* of the partial solution Π in P. Observe that the inner vector $\mathbf{a}$ and the outer vector $\mathbf{b}$ are compatible. As $\mathbf{a} \in \overline{\mathcal{R}}_\tau$ and $\overline{\mathcal{R}}'_\tau$ is a representative subset of $\overline{\mathcal{R}}_\tau$, there is an $\mathbf{a}' \in \overline{\mathcal{R}}_\tau$ that is also compatible with $\mathbf{b}$. Thus there is a partial solution $\Pi' \in \mathcal{R}'_\tau \subseteq \mathcal{R}'$ having inner vector $\mathbf{a}'$.

We claim that replacing Π' at (A, B) in P gives a valid solution P'. If a path P of Q has both endpoints outside $V(A)$, then there is a corresponding valid path after the modification: as the two partial solutions have the same type, the set S_0 and the matching M are the same in both of them. Therefore, whenever P has an $x - y$ subpath in A for some $x, y \in S$, then this subpath is a path of class (C0) or (C2) in Π, hence there is a path with the same endpoints in Π'. If a path of Q has one endpoint in $V(A)$, then the other endpoint is outside $V(A)$ (as H has no edge in $V(A)$). Therefore, the endpoint of Q in $V(A)$ is the endpoint of a path of class (C1) of Π. Suppose that this path connects a_j to $v_j \in J$ and the other endpoint of the path Q is b_j. As the inner vector $\mathbf{a}'$ of Π' is compatible with outer vector $\mathbf{b}$, we get that a'_j and b_j are adjacent in H. It follows that P' contains a valid path from a'_j to b_j (note that this path may reenter $V(A)$ several times, thus we need to use again that S_0 and M are the same in both partial solutions).

We have shown that Π' is replacable in P, resulting in a solution P'. Thus we have shown that $\mathcal{R}'$ is a representative set of partial solutions. □

We are now able to present the proof of Lemma 3.3.

Proof. (of Lemma 3.3) Let $r^* = \max\{r, 10k^2\}$ and let $I^{\text{sep}}_{k,r,z} := k \cdot R_{z,r^*}$, where R_{z,r^*} is the constant in Lemma 3.7. We invoke the algorithm of Lemma 3.7 on the separation (A, B). If it returns an induced matching of size r^* or an induced skew biclique on $r^* + r^*$ vertices, then we are done (as $r^* \geq r$). If Lemma 3.7 returns a clique of size $r^* \geq 10k^2$, then we invoke Lemma 3.2, which either returns k-disjoint valid paths or an irrelevant terminal; we are done in both cases. Otherwise, let $\mathcal{R}$ be the representative set of size at most R_{z,r^*} returned by the algorithm of Lemma 3.7. Each partial solution of $\mathcal{R}$ uses at most k terminals of $V(A) \cap T$ as endpoints. Therefore, if we let T^* contain every terminal that is an endpoint of a path in one of the partial solutions in $\mathcal{R}$, then we have $|T^*| \leq k|\mathcal{R}| \leq k \cdot R_{z,r^*} = I^{\text{sep}}_{k,r,z}$. The assumption $|V(A) \cap T| > I^{\text{sep}}_{k,r,z}$ implies that there is a $t \in (V(A) \cap T) \setminus T^*$. We claim that removing t from the set of terminals does not change the solvability of the instance. Let P be a solution and let Π be its partial solution at (A, B). If t is not the endpoint of path in P, then the solution remains a valid even after removing t from the set of terminals. Otherwise, as $\mathcal{R}$ is representative, there is a partial solution $\Pi' \in \mathcal{R}$ that is replacable into P; let P' be the resulting solution. By the definition, if P' has a path ending in $V(A) \cap T$, then this terminal is endpoint of a path in Π' and hence in T^*. Therefore, $t \notin T^*$ is not the endpoint of any of the paths in P'. This means that P' is a valid solution after removing t from the set of terminals and hence t is an irrelevant terminal. □

3.4 Reducing the number of components

With repeated applications of Lemma 3.3, we can reduce the number of terminals in each component to at most a constant $I^{\text{sep}}_{k,r,z}$. The final step of the algorithm is to reduce the number of components that contain terminals. (We remark that it would be possible to reduce also the number of components *not* having any terminals at all, as their only role is to provide connectivity to Z, but we do not need this stronger claim here.)

Lemma 3.8. *Let $Z \subseteq V(G)$ be a set of vertices disjoint from T such that for every component C of $G-Z$, we have $|T \cap V(C)| \le q$ and the set $T \cap V(C)$ is independent in H. If $|T| > 100|Z|^4q^2$, then we can identify an irrelevant terminal in polynomial time.*

Proof. For every ordered pair (z_1, z_2) of vertices in Z (possibly with $z_1 = z_2$), we mark some of the terminals. We proceed the following way for the pair (z_1, z_2). Let $\mathcal{T}_{(z_1,z_2)}$ contain every ordered pair (t_1, t_2) of terminals with the following properties:

- t_1 and t_2 are adjacent in H.
- There is a $t_1 - z_1$ path whose internal vertices are disjoint from Z.
- There is a $t_2 - z_2$ path whose internal vertices are disjoint from Z.

Clearly, the collection $\mathcal{T}_{(z_1,z_2)}$ can be constructed in polynomial time. Note that by the requirement that t_1 and t_2 are adjacent in H, we have that t_1 and t_2 are in different components of $G-Z$ for every $(t_1, t_2) \in \mathcal{T}_{(z_1,z_2)}$.

Let $b = 2|Z|q + 1$. First, let us select greedily a maximal collection of pairs from $\mathcal{T}_{(z_1,z_2)}$ such that every terminal appears in at most one select pair. If we find b such pairs, then we mark the (exactly) $2b$ terminals appearing in these pairs and we are done with processing (z_1, z_2). If we do not find b such pairs, then this means that we can find a set X of at most $2(b-1)$ terminals such that every pair of $\mathcal{T}_{(z_1,z_2)}$ contains a terminal from X (either at the first or second coordinate). Let us mark every terminal in X. Furthermore, for every $u \in X$, let us mark b terminals t^* such that $(t^*, u) \in \mathcal{T}_{(z_1,z_2)}$ (or all of them if there are less than b such terminals). This completes the description of the marking procedure. We are considering $|Z|^2$ pairs (z_1, z_2) and for each pair, we mark at most $\max\{2b, 2(b-1)\cdot(b+1)\} = 2(b-1)\cdot(b+1)$ terminals. Therefore, if there are more than $100|Z|^4q^2 > |Z|^2 \cdot 2(b-1)(b+1)$ terminals, then there is a unmarked terminal. We claim that any unmarked terminal is irrelevant.

Let t be an unmarked terminal and consider a solution to the instance where t is the endpoint of a path P of the solution; let u be the other endpoint of P. By assumption, $G-Z$ has no valid path and Z is disjoint from T, thus path P contains at least one vertex of Z. Starting at v, let z_1 and z_2 be the first and last vertices of P in Z, respectively (it is possible that $z_1 = z_2$). Then path P shows that (t, u) appears in the collection $\mathcal{T}_{(z_1,z_2)}$. Consider first the case when the marking procedure for (z_1, z_2) found b pairs not sharing any terminals. Observe that the paths of the solution intersect at most $2|Z|$ components of $G-Z$: each path contains at least one vertex of Z and these $|Z|$ vertices can break the paths of the solution into at most $2|Z|$ subpaths. This means that there are at most $2|Z|q$ terminals that are in a component of $G-Z$ intersected by the solution. Therefore, as we have found $b = 2|Z|q+1$ pairs, there is a pair (t_1, t_2) among them such that the components of t_1 and t_2 in $G-Z$ are disjoint from the solution. As $(t_1, t_2) \in \mathcal{T}_{(z_1,z_2)}$, the definition of $\mathcal{T}_{(z_1,z_2)}$ implies that t_1 and t_2 are adjacent in H (which means that they are in different components of $G-Z$). Furthermore, for $i = 1, 2$, we can choose a $t_i - z_i$ path P_i whose internal vertices are disjoint from Z. This means that the internal vertices of P_i are in the same component of $G-Z$ as t_i, implying that they are disjoint from the solution. We modify the solution: we replace the $v-z_1$ subpath of P with the $t_1 - z_1$ path P_1 and the $z_2 - u$ subpath of P with the $z_2 - t_2$ path P_2. This gives a valid $t_1 - t_2$ path that is disjoint from every other path in the solution. Therefore, we have found a solution not involving the terminal t.

Consider now the case when the marking procedure did not find b pairs and hence found a set X of at most $2(b-1)$ terminals. As $(t, u) \in \mathcal{T}_{(z_1,z_2)}$, either t or u is in X. If t is in X, then we marked t, thus let us assume that u is in X. Then we marked some terminals t^* for which (t^*, u) is in $\mathcal{T}_{(z_1,z_1)}$. If t itself was not marked this way, then we marked $b = 2|Z|+1$ such terminals t^*. As the solution intersects at most $2|Z|$ components of $G-Z$ and each component contains at most q terminals, there is a marked terminal t^* whose component is disjoint from the solution and (t^*, u) is in $\mathcal{T}_{(z_1,z_2)}$. By the definition of $\mathcal{T}_{(z_1,z_2)}$, this means that t^* and u are adjacent and the component of t^* is adjacent to z_1. Let us choose a $t^* - z_1$ path P^* whose internal vertices are in the component of t^* in $G-Z$ (and hence disjoint from the solution). Let us modify the path P by replacing the $t - z_1$ subpath with the $t^* - z_1$ subpath P. This way, we obtain a solution not involving the terminal t also in this case, showing that t is indeed irrelevant. □

We are now ready to prove Lemma 3.1.

Proof. (of Lemma 3.1) Let $I_{k,r,z} := 100z^4(I^{\text{sep}}_{k,r,z})^2$. Suppose first that a component C of $G-Z$ contains more than $I^{\text{sep}}_{k,r,z}$ terminals. Then let (A, B) be the separation of G with $V(A) \setminus V(B) = C$ and let us invoke the algorithm of Lemma 3.3. It either returns an irrelevant terminal, a solution with k paths, an induced matching in H, or an induced skew biclique in H; in all cases, we are done. Assume therefore that every component C of $G - Z$ contains at most $I^{\text{sep}}_{k,r,z}$ terminals. Then the algorithm of Lemma 3.8 gives an irrelevant terminal. □

References

[1] E. Birmelé, A. Bondy, and B. Reed. The Erdős-Pósa property for long circuits. *Combinatorica*, 27:135–145, 2007.

[2] R. H. Chitnis, M. Hajiaghayi, and G. Kortsarz. Fixed-parameter and approximation algorithms: A new look. In *IPEC*, pages 110–122, 2013.

[3] R. Diestel, K.-i. Kawarabayashi, and P. Wollan. The Erdős-Pósa property for clique minors in highly connected graphs. *J. Combin. Theory Ser. B*, 102(2):92–114, 2012.

[4] P. Erdős and L. Pósa. On independent circuits contained in a graph. *Canad. J. Math.*, 17:347–352, 1965.

[5] S. Even, A. Itai, and A. Shamir. On the complexity of time table and multi-commodity flow problems. In *SFCS '75*, SFCS '75, pages 184–193, Washington, DC, USA, 1975. IEEE Computer Society.

[6] F. V. Fomin, D. Lokshtanov, F. Panolan, and S. Saurabh. Representative sets of product families. *CoRR*, abs/1402.3909, 2014.

[7] F. V. Fomin, D. Lokshtanov, and S. Saurabh. Efficient computation of representative sets with applications in parameterized and exact algorithms. In *SODA*, pages 142–151, 2014.

[8] T. Gallai. Maximum-minimum Sätze und verallgemeinerte Faktoren von Graphen. *Acta Math. Acad. Sci. Hungar.*, 12:131–173, 1961.

[9] M. Grohe and M. Grüber. Parameterized approximability of the disjoint cycle problem. In *ICALP*, pages 363–374, 2007.

[10] H. Hirai and G. Pap. Tree metrics and edge-disjoint S-paths. *Mathematical Programming*, pages 1–43, 2013.

[11] K.-i. Kawarabayashi and P. Wollan. Non-zero disjoint cycles in highly connected group labelled graphs. *J. Comb. Theory Ser. B*, 96(2):296–301, Mar. 2006.

[12] K.-i. Kawarabayashi and P. Wollan. A shorter proof of the graph minor algorithm: The unique linkage theorem. In *STOC '10*, pages 687–694, New York, NY, USA, 2010. ACM.

[13] B. Korte, L. Lovász, H. J. Prömel, and A. Schrijver, editors. *Paths, flows, and VLSI-layout*, volume 9 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 1990. Papers from the meeting held at the University of Bonn, Bonn, June 20–July 1, 1988.

[14] L. Lovász. Matroid matching and some applications. *Journal of Combinatorial Theory, Series B*, 28(2):208 – 236, 1980.

[15] W. Mader. Uber die Maximalzahl kantendisjunkter A-Wege. *Archiv der Math*, 30(2):325–336, 1978.

[16] D. Marx. Parameterized coloring problems on chordal graphs. *Theor. Comput. Sci.*, 351(3):407–424, 2006.

[17] D. Marx. Parameterized complexity and approximation algorithms. *The Computer Journal*, 51(1):60–78, 2008.

[18] D. Marx. A parameterized view on matroid optimization problems. *Theor. Comput. Sci.*, 410(44):4471–4479, 2009.

[19] D. Marx and I. Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM J. Comput.*, 43(2):355–388, 2014.

[20] M. Middendorf and F. Pfeiffer. On the complexity of the disjoint paths problem. *Combinatorica*, 13(1):97–107, 1993.

[21] B. Monien. How to find long paths efficiently. In *Analysis and design of algorithms for combinatorial problems (Udine, 1982)*, volume 109 of *North-Holland Math. Stud.*, pages 239–254. North-Holland, Amsterdam, 1985.

[22] G. Naves and A. Seb. Multiflow feasibility: An annotated tableau. In W. Cook, L. Lovsz, and J. Vygen, editors, *Research Trends in Combinatorial Optimization*, pages 261–283. Springer Berlin Heidelberg, 2009.

[23] D. Rautenbach and B. Reed. The Erdős-Pósa property for odd cycles in highly connected graphs. *Combinatorica*, 21:267–278, 2001.

[24] B. Reed. Mangoes and blueberries. *Combinatorica*, 19:267–296, 1999.

[25] B. Reed, N. Robertson, P. Seymour, and R. Thomas. Packing directed circuits. *Combinatorica*, 16(4):535–554, 1995.

[26] N. Robertson and P. Seymour. Graph minors. V. Excluding a planar graph. *J. Combin. Theory Ser. B*, 41(1):92–114, 1986.

[27] N. Robertson and P. Seymour. Graph minors. X. Obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153 – 190, 1991.

[28] N. Robertson and P. D. Seymour. Graph minors. XIII. The disjoint paths problem. *J. Combin. Theory Ser. B*, 63(1):65–110, 1995.

[29] H. Shachnai and M. Zehavi. Faster computation of representative families for uniform matroids with applications. *CoRR*, abs/1402.3547, 2014.

[30] C. Thomassen. On the presence of disjoint subgraphs of a specified type. *J. Graph Theory*, 12:101–110, 1988.

[31] C. Thomassen. The Erdős-Pósa property for odd cycles in graphs with large connectivity. *Combinatorica*, 21:321–333, 2001.

[32] P. Wollan. Packing cycles with modularity constraints. *Combinatorica*, 31(1):95–126, 2011.