Part III

Parity Based Graph Separation
7

Parity Multiway Cut

7.1 Introduction

In this chapter, we study a parity based generalization of the classical Multiway Cut problem. Formally, we study the Parity Multiway Cut problem which is defined as:

**Parity Multiway Cut (PMWC)**

**Parameter**: \( k \)

**Input**: A graph \( G = (V, E) \), vertex subsets \( T_e \) and \( T_o \) \((T = T_e \cup T_o)\), integer \( k \)

**Question**: Is there a vertex set \( S \) of size at most \( k \) which intersects

1. all odd paths from a vertex \( v \in T_o \) to every other vertex \( u \in T \setminus \{v\} \),
2. all even paths from a vertex \( v \in T_e \) to every other vertex \( u \in T \setminus \{v\} \)?

When \( T_e = T_o \), this is precisely the classical Multiway Cut problem. If \( T_o = \emptyset \) then this is the Even Multiway Cut problem and if \( T_e = \emptyset \) then this is the Odd Multiway Cut problem.

Unlike Multiway Cut, PMWC (when the solution is not allowed to contain the
terminals) is already \textbf{NP}-complete for the case when $|T| = 2$. Indeed, consider the following reduction from \textsc{Vertex Cover} to \textsc{PMWC}. Given an instance $(G = (V, E), k)$ of \textsc{Vertex Cover}, add two new vertices $t_1$ and $t_2$, make them both adjacent to every vertex in $V$, and set $T_o = \{t_1, t_2\}$ and $T_e = \emptyset$. Call this new graph $G'$. It is easy to see that $G$ has a vertex cover of size at most $k$ if and only if $G'$ has $k$-sized vertex subset that intersects every odd $T_o$-path. In fact, our argument shows that \textsc{OMWC} is \textbf{NP}-complete for the case when $|T| = 2$. One can similarly show that \textsc{EMWC} is \textbf{NP}-complete for the case when $|T| = 2$.

In this chapter, we study the variant of the problem where the solution to an instance $(G, T = T_e \cup T_o, k)$ of \textsc{PMWC} is allowed to intersect the set $T$. This problem is \textbf{NP}-complete for unbounded $T$ because it is a generalization of \textsc{Multiway Cut}. In this chapter, the focus is on optimizing the dependence of our algorithms on the parameter $k$. Therefore, we will suppress the dependence of the algorithms on the input size by using the $O^*(\cdot)$ notation. Our main result with respect to this problem is the following.

**Theorem 7.1.1.** \textsc{PMWC} can be solved in time $O^*(2^{O(k^3)})$.

In the course of obtaining our algorithm, we introduce two generic subroutines which apply to a wider range of graph separation problems. In particular,

- we first design an \textbf{FPT} algorithm that reduces the input instance to a highly structured instance while preserving a solution. In this particular case, the structure induced on the given instance is bipartiteness of the input graph. This algorithm gives a general approach for parity based graph separation problems and has already been utilized for other problems as well.

- we introduce a generalization of important separators which allows us to “overload” certain properties onto the parts of the graph disjoint from the separator.
The motivation behind such a generalization is that it allows us to define a more involved notion of dominating set of separators and therefore paves the way for the application of the Important Separators Template.

**Overview of the algorithm.** The algorithm for PMWC has three stages; in the first stage, using the technique of iterative compression, we reduce the problem to solving a bounded number of instances of the problem where the even terminals are bounded by a linear function of \( k \). In the second stage, we reduce the instance to one with a solution whose removal leaves a bipartite graph. In the final stage, we define the generalization of important separators and apply the important separator template to solve this restricted version of the problem.

We note that the special case OMWC can be shown to be FPT be a reduction to the Subset Odd Cycle Transversal problem which was shown to be FPT in [58]. However, such an algorithm for OMWC would have a significantly worse dependence on the parameter \( k \) when compared to the algorithm we present in this chapter.

Our main motivation for studying this parity based generalizations of graph separation problems is the recently initiated parameterized study of parity versions of graphs minors by Kawarabayashi, Reed and Wollan [55] and separation problems similar to Multiway Cut [10, 19, 79]. Furthermore, Geelen et al. [40] proved an odd variant of Mader’s \( T \)-path Theorem in which they showed that, given a graph \( G \) and a subset \( T \) of vertices, there are either \( k \) vertex disjoint odd \( T \)-paths, or there is a vertex set of size at most \( 2k \) which intersects every odd \( T \)-path. This result has already turned out to be useful in graph theory [40, 59], as well as in the design of parameterized algorithms [52, 54, 53]. This result was crucial in settling the parameterized complexity of finding \( k \) vertex disjoint odd length cycles in a graph [53]. Observe that, this odd variant of Mader’s \( T \)-path Theorem naturally gives rise to the OMWC problem, which is a
special case of PMWC.

In an instance \((G, T = T_e \cup T_o, k)\) of PMWC, the vertices in \(T\) are called terminals, those in \(T_e\) are called even terminals and those in \(T_o\) are called odd terminals. Vertices in \(T_e \setminus T_o\) are called purely even terminals and those in \(T_o \setminus T_e\) are called purely odd terminals.

### 7.2 Bounding the number of even terminals

In this section, we give an algorithm which allows us to assume that the number of even terminals in a given instance is bounded linearly in \(k\).

#### 7.2.1 Preprocessing

We begin with a preprocessing rule which allows us to assume that the solution we are searching for is disjoint from the set of terminals.

**Preprocessing Rule 1.** Let \((G, T = T_e \cup T_o, k)\) be an instance of PMWC and let \(T = \{t_1, \ldots, t_l\}\). For every terminal \(t_i \in T\), add 2 new vertices \(t^1_i\) and \(t^2_i\) and add edges \((t^1_i, t^1_i)\) and \((t^1_i, t^2_i)\). Let this graph be \(G'\). Set \(T'_o = \{t^2_i | t_i \in T_o\}\) and \(T'_e = \{t^2_i | t_i \in T_e\}\) and \(T' = T'_o \cup T'_e\). Finally, return the instance \((G', T' = T'_e \cup T'_o, k)\).

It is clear that the above rule can be applied in polynomial time. The correctness of the rule follows from the following lemma.

**Lemma 7.2.1.** Given an instance \((G, T = T_e \cup T_o, k)\) of PMWC, let \((G', T' = T'_e \cup T'_o, k)\) be the instance obtained by the application of Preprocessing Rule 1 on \((G, T = T_e \cup T_o, k)\). Then, \((G, T = T_e \cup T_o, k)\) is a YES instance if and only if \((G', T' = T'_e \cup T'_o, k)\) is a YES instance and has a solution disjoint from \(T'\).
Proof. Suppose that $S$ is a solution for the instance $(G, T = T_e \cup T_o, k)$. We claim that $S$ is a solution for the instance $(G', T' = T_e \cup T'_o, k)$ disjoint from $T'$. Clearly, $|S| \leq k$ and $S$ is disjoint from $T'$. If $S$ were not a solution for the instance $(G', T' = T'_e \cup T'_o, k)$ then there is a path $P$ of forbidden parity between two terminals $t^2_i$ and $t^2_j$. But the subpath of $P$ from $t_i$ to $t_j$ is a path with the same parity as $P$ from $t_i$ to $t_j$ in $G \setminus S$, a contradiction.

Conversely, suppose that $S'$ is a solution for the instance $(G', T', k)$ disjoint from $T'$. If $S'$ contains a vertex not in $G$, then it must be a vertex $t^1_i$ for some $i$. For every $i$ such that $t^1_i \in S'$, we replace it with the vertex $t_i$. Let the resulting set be $S$. Now, the vertices in $S$ are all present in $G$. We claim that $S$ is a solution for the instance $(G, T = T_e \cup T_o, k)$. Clearly, $|S| \leq k$. Therefore, if $S$ were not a solution, then there is a path $P$ of forbidden parity between two terminals $t_i$ and $t_j$ in the graph $G \setminus S$. Since $t_i$ and $t_j$ are not in $S$, $t^1_i$ and $t^1_j$ are not in $S'$. Let $P_1$ be the path comprising two edges from $t^2_i$ to $t_i$, that is, $P_1 = t^2_i, t^1_i, t_i$ and let $P_2$ be the path comprising two edges from $t_j$ to $t^2_j$, that is, $P_2 = t_j, t^1_j, t^2_j$. Since $P$ is a path in $G$, the paths $P$ and $P_1$ intersect exactly in $t_i$ and the paths $P$ and $P_2$ intersect exactly in $t_j$. Therefore, the path $P_1 + P + P_2$ is a path from $t^2_i$ to $t^2_j$ in $G' \setminus S'$. However, since the paths $P_1$ and $P_2$ are of length 2, the path $P_1 + P + P_2$ is a path of the same parity as $P$ from $t^2_i$ to $t^2_j$ in $G' \setminus S'$, a contradiction. This completes the proof of the lemma. 

Due to the above preprocessing rule, we assume without loss of generality that the given instance of PMWC is such that each terminal has exactly 1 neighbor. Furthermore, we also assume that the objective is to check if there is a pmwc for the given instance disjoint from the terminal set. Formally, we redefine the PMWC problem in the following form.
Parity Multiway Cut (PMWC)  

**Parameter:** $k$

**Input:** A graph $G = (V, E)$, vertex subsets $T_e$ and $T_o$ ($T = T_e \cup T_o$), integer $k$

**Question:** Is there a vertex set $S$ of size at most $k$ which is disjoint from $T$ and intersects

1. all odd paths from a vertex $v \in T_o$ to every other vertex $u \in T \setminus \{v\}$,
2. all even paths from a vertex $v \in T_e$ to every other vertex $u \in T \setminus \{v\}$?

Any set which hits all odd paths from a vertex $v \in T_o$ to every other vertex $u \in T \setminus \{v\}$ and all even paths from a vertex $v \in T_e$ to every other vertex $u \in T \setminus \{v\}$ is referred to as a pmwc for the given instance. We also initially perform the following preprocessing step on the given instance $(G, T = T_e \cup T_o, k)$ of PMWC.

**Preprocessing Rule 2.** Given an instance $(G, T = T_e \cup T_o, k)$ of PMWC, if there is a vertex $v$ which does not lie on a $T$-path, then return the instance $(G \setminus \{v\}, T = T_e \cup T_o, k)$.

It is clear that the above rule is correct and can be applied in polynomial time by testing for every vertex if it lies on a $T$-path. Henceforth, we assume that Preprocessing Rule 2 is not applicable on the given input instance, which implies that every vertex lies on a $T$-path.

### 7.2.2 Reducing even terminals

We first describe a way to reduce the given instance of PMWC to multiple (but a bounded number of) instances, each with a bounded number of even terminals, such that solving these instances will lead to a solution for the input instance. To do this, we
begin by using the technique of iterative compression.

Given an instance \((G = (V, E), T = T_e \cup T_o, k)\) of PMWC, where \(V = \{v_1, \ldots, v_n\}\), we define the graph \(G_i\) as \(G_i = G[V_i]\) where \(V_i = \{v_1, \ldots, v_i\}\). We iterate through the instances \((G_i, T_i = (T_e \cap V_i) \cup (T_o \cap V_i), k)\) starting from \(i = k + 1\) and for the \(i^{th}\) instance, with the help of a known solution \(S_i\) of size at most \(k + 1\) we try to find a solution \(\hat{S}_i\) of size at most \(k\). Formally, the compression problem we address is following.

<table>
<thead>
<tr>
<th>PMWC COMPRESSION</th>
<th>Parameter: (k)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> An instance ((G, T = T_e \cup T_o, k)) of PMWC, and a pmwc (S) of size at most (k + 1).</td>
<td></td>
</tr>
<tr>
<td><strong>Question:</strong> Is there a pmwc for the instance ((G, T = T_e \cup T_o, k)) of size at most (k)?</td>
<td></td>
</tr>
</tbody>
</table>

We reduce the PMWC problem to \(n - k\) instances of the PMWC COMPRESSION problem as follows. Let \(I_i = (G_i, (T_e \cap V_i) \cup (T_o \cap V_i), S_i, k)\) be the \(i^{th}\) instance of PMWC COMPRESSION. The set \(V_{k+1}\) is clearly a pmwc of size at most \(k + 1\) for the instance \(I_{k+1}\). It is also easy to see that if \(\hat{S}_{i-1}\) is a pmwc of size at most \(k\) for instance \(I_{i-1}\), then the set \(\hat{S}_{i-1} \cup \{v_i\}\) (or the set \(\hat{S}_{i-1} \cup N(v_i)\) if \(v_i\) is a terminal) is a pmwc of size at most \(k + 1\) for the instance \(I_i\). We use these two observations to start off the iteration with the instance \((G_{k+1}, (T_e \cap V_{k+1}) \cup (T_o \cap V_{k+1}), k, S_{k+1})\) where \(S_{k+1} = V_{k+1}\) and check if there is a solution for this instance. If there is such a solution \(\hat{S}_{k+1}\), we set \(S_{k+2} = \hat{S}_{k+1} \cup \{v_{k+2}\}\) (or \(S_{k+2} = \hat{S}_{k+1} \cup N(v_{k+2})\) if \(v_{k+2}\) is a terminal) and try to compute a pmwc of size at most \(k\) for the instance \(I_{k+2}\) and so on. If, during any iteration, the corresponding instance is found to be a NO instance then it implies that the original instance is also a NO instance. Finally the solution for the original input instance is the set \(\hat{S}_n\). Since there can be at most \(n\) such iterations, the total time taken is bounded by
Observation 7.2.2. Let \((G, T = T_o \cup T_e, k)\) be an instance of PMWC and let \(S\) be a solution for this instance.

(a) Consider a connected component of \(G \setminus S\) which contains at least two terminals. Such a component cannot contain terminals from both \(T_o\) and \(T_e\).

(b) Any connected component of \(G \setminus S\) contains at most 2 vertices from \(T_e\).

Proof. (a) Let \(C\) be a component of \(G \setminus S\) containing two terminals \(t_1\) and \(t_2\) such that \(t_1 \in T_o\) and \(t_2 \in T_e\). Then, there is a path between \(t_1\) and \(t_2\) in \(G \setminus S\). If this path is odd, then it contradicts the intersection of \(S\) with every odd \(t_1-T_o \setminus \{t_1\}\) path and if it is even, then it contradicts the intersection of \(S\) with every even \(t_2-T_e \setminus \{t_2\}\) path.

(b) Let \(C\) be a component of \(G \setminus S\) containing a set \(T'_e \subseteq T_e\). Consider a tree spanning the set \(T'_e\) in \(G \setminus S\). Since \(T'_e\) lies in a connected component of \(G \setminus S\), such a tree exists. Since this tree is a connected bipartite graph, if \(|T'_e| > 2\), there must be two vertices of \(T'_e\) with an even path between them. But this is a contradiction since this even path is disjoint from \(S\).

We now show that if an instance \((G, T, k, S)\) of PMWC COMPRESSION has a solution disjoint from some pmwc \(S\), then it must be the case that the number of even terminals in this instance is bounded linearly in \(k\). Formally,

Lemma 7.2.3. Consider an instance \((G, T = T_o \cup T_e, k, S)\) of PMWC COMPRESSION and suppose that this instance has a solution disjoint from the pmwc \(S\). Then, \(|T_e| \leq 6k\).

Proof. Let \(S'\) be the solution disjoint from \(S\). We call a component of \(G \setminus S\) affected if it contains a vertex of \(S'\) and unaffected otherwise. Clearly, there can be at most \(k\) affected components in \(G \setminus S\). Now, consider the unaffected components of \(G \setminus S\) which contain
an even terminal. We claim that the number of such components does not exceed $2k$. If this were not the case, then there exist three unaffected components which contain an even terminal each and a vertex $v \in S$ such that the three unaffected components are adjacent $v$. But this implies the presence of an even path between at least two of these even terminals which is disjoint from the solution $S'$, a contradiction. Hence, the number of components of $G \setminus S$ which contain an even terminal is at most $3k$ (at most $k$ affected and at most $2k$ unaffected components). By Observation 7.2.2, any component of $G \setminus S$ can contain at most 2 even terminals. Since $S$ is disjoint from $T_e$, the number of even terminals in the instance is bounded by $2 \cdot 3k = 6k$. This completes the proof of the lemma.

We now describe a way to bound the number of even terminals in an instance of PMWC COMPRESSION. Formally,

Lemma 7.2.4. There is an algorithm that, given an instance $I = (G, T = T_e \cup T_o, k, S)$ of PMWC COMPRESSION, runs in time $O^*(2^{O(k)})$ and returns $2^{k+1}$ instances of PMWC $\{(G_i, T_i = T_{ei} \cup T_{oi}, k_i)\}_{1 \leq i \leq \ell}$ where $\ell = 2^{k+1}$ such that

1. $(G, T, k, S)$ is a YES instance of PMWC COMPRESSION if and only if there is an $1 \leq i \leq \ell$ such that $(G_i, T_i, k_i)$ is a YES instance of PMWC.

2. For each $1 \leq i \leq \ell$, $k_i \leq k$ and $|T_{ei}| \leq 6k_i$.

Proof. For every $S' \subseteq S$, we obtain an instance $I_{S'}$ of PMWC COMPRESSION corresponding to $S'$ by deleting $S'$ from the instance $I$ and applying Preprocessing Rule 2 exhaustively on the resulting instance. Thus, we obtain $2^{k+1}$ instances of PMWC COMPRESSION, each corresponding to a subset of $S$.

1. Suppose that $(G, T, k, S)$ is a YES instance of PMWC COMPRESSION. We fix a hypothetical solution $\hat{S}$ for this instance. Let $Y = S \cap \hat{S}$ and consider the instance
\[ I_Y = (G \setminus Y, T \setminus Y, k - |Y|) \]. Then, \( I_Y \) is a \textbf{Yes} instance of PMWC and \( \hat{S} \setminus Y \) is a solution for this instance which is also disjoint from \( S \setminus Y \).

Conversely, if \((G, T, k, S)\) were a \textbf{No} instance, then it follows that for every \( S' \subseteq S \), the corresponding instance \( I_{S'} = (G \setminus S', T \setminus S', k - |S'|) \) of PMWC is also a \textbf{No} instance.

2. Observe that if \((G, T, k, S)\) is a \textbf{Yes} instance and \( Y = S \cap \hat{S} \) then the instance \( I_Y \) has a solution disjoint from \( S \setminus Y \) and by Lemma 7.2.3, the number of even terminals in this instance cannot exceed \( 6(k - |Y|) \). Therefore, if for any instance \( I_{S'} \), the number of even terminals in this instance exceeds \( 6(k - |S'|) \), then we term the instance invalid and reject it. Therefore, we may assume that at this juncture, we have at most \( 2^{k+1} \) instances of PMWC and the number of even terminals in each instance \( I_{S'} \) is bounded by \( 6(k - |S'|) \). The algorithm finally returns these instances. We have already shown that these instances satisfy the properties in the statement of the lemma. Since each such instance can be constructed in polynomial time and there are at most \( 2^{k+1} \) such instances, the algorithm runs in time \( O^*(2^{O(k)}) \). This completes the proof of the lemma.

Henceforth, we assume that the given PMWC instance is one returned by the algorithm of Lemma 7.2.4.

### 7.3 Obtaining a bipartition instance

In this section, we give the first of our subroutines which imposes a certain structure on the given graph. This will be described and proved formally in the rest of this section.
7.3.1 Isolated and semi-isolated components

**Definition 7.3.1.** Consider an instance \((G, T = T_e \cup T_o, k)\) of PMWC and let \(S\) be a solution for this instance (see Fig. 7.1). A connected component of \(G \setminus S\) is called an **isolated component** in \(G \setminus S\) if it is disjoint from the set of terminals and a **non-isolated component** (in \(G \setminus S\)) otherwise. Vertices in an isolated component are called isolated vertices and those in a non-isolated component are called non-isolated vertices.

**Definition 7.3.2.** Consider an instance \((G, T = T_e \cup T_o, k)\) of PMWC and let \(S\) be a solution for this instance. Consider a non-isolated component \(C\) in \(G \setminus S\) which contains at least 2 terminals. We define a **main component** of \(C\) as a maximal connected subgraph of \(G[C]\) such that every vertex in this subgraph lies on a \(T\)-path in \(G \setminus S\). Consider a non-isolated component of \(C\) in \(G \setminus S\) which contains a single terminal \(t \in T\). We know that \(t\) has a single neighbor in \(G\), say \(u\). We define the main component of \(C\) as the edge \((u, t)\).

**Observation 7.3.3.** Consider an instance \((G, T = T_e \cup T_o, k)\) of PMWC and let \(S\) be a solution for this instance. Consider a non-isolated component \(C\) in \(G \setminus S\). Then, there is a unique main component of \(C\).

**Proof.** If \(C\) contains a single terminal, it is clear that there is a unique main component by definition. Therefore, we assume that \(C\) contains at least 2 terminals. Suppose that there were two distinct main components of \(C\), \(M_1\) and \(M_2\). Since each main component is a maximal connected subgraph, the two main components must be disjoint and non-adjacent. Furthermore, each main component contains a terminal. Since \(C\) is a connected component of \(G \setminus S\), there is a path \(P\) in \(G[C]\) from \(M_1\) to \(M_2\). Recall that \(P\) is not an edge. Since both main components contain terminals, there is path from a terminal in \(M_1\) to a terminal in \(M_2\) which contains \(P\) as a subpath. However, this implies

143
that every vertex in $P$ lies on a $T$-path in $G \setminus S$, which contradicts the maximality of $M_1$ and $M_2$. Hence, we conclude that there is a unique main component of $C$. 

**Definition 7.3.4.** Consider an instance $(G, T = T_e \cup T_o, k)$ of PMWC and let $S$ be a solution for this instance. Consider a non-isolated component $C$ in $G \setminus S$ and let $M$ be the main component of $C$. Then, the connected components of the graph $G[C \setminus M]$ are called the **semi-isolated components** of $C$ in $G \setminus S$. Vertices in a semi-isolated component are called **semi-isolated vertices**.

**Observation 7.3.5.** Consider an instance $(G, T = T_e \cup T_o, k)$ of PMWC and let $S$ be a solution for this instance. Consider a non-isolated component $C$ in $G \setminus S$ and let $M$ be the main component of $C$ in $G \setminus S$. Every terminal in $C$ is also present in $M$. Furthermore, if $C'$ is a semi-isolated component of $C$, then there is no $T$-path in $G \setminus S$ which intersects $C'$.

**Proof.** Both statements are clearly true when $C$ contains a single terminal. Therefore, we assume that $C$ contains at least 2 terminals. Now, let $T' = T \cap C$ and consider a minimal tree, say $H$, in $G[C]$ which spans the terminals in $T'$. Since $C$ is connected component, such a tree exists. Since $H$ is minimal, every vertex in $H$ lies on a $T$-path within $H$. Since the main component is unique, $H$ is contained in the main component, which implies that all the vertices in $T'$ occur in the main component of $C$. Furthermore, if there is a semi-isolated component $C'$ such that there is a $T'$-path in $C$ intersecting a vertex in $C'$ then we may simply add the vertices of this $T'$-path to the main component to get another connected subgraph with every vertex lying on a $T$-path in $G \setminus S$. Since the main component is strictly contained inside this subgraph, it contradicts the maximality of the main component. 

144
Lemma 7.3.6. Let \((G, T = T_e \cup T_o, k)\) be an instance of PMWC and let \(S\) be a solution for this instance. Then, any semi-isolated component has exactly 1 neighbor in the adjacent main component.

Proof. Consider a semi-isolated component \(C'\) and let \(M\) be the main component adjacent to \(C'\). We will now show that \(C'\) cannot have more than one neighbor in \(M\).

Suppose that this is not the case and let \(v_1\) and \(v_2\) be two distinct vertices in \(M\) which are adjacent to vertices in \(C'\). We note that if \(v_1\) and \(v_2\) are terminals, then there is a path between these two terminals which intersects the semi-isolated component \(C'\), which is not possible by Observation 7.3.5. Now, consider the case when exactly one of the two vertices, say \(v_1\), is a terminal. But, \(v_2\) lies on a path between two terminals, say \(w_1\) and \(w_2\) where \(w_1\) or \(w_2\) could be \(v_1\). Consider this path and consider the two subpaths of this path from \(v_2\) to \(w_1\) and from \(v_2\) to \(w_2\). At least one of these two subpaths is disjoint from \(v_1\). Hence, this subpath, along with the edges from \(C'\) to \(v_1\) and \(v_2\) results in a \(T\)-path..
which intersects a semi-isolated component, a contradiction.

We now consider the remaining case where \( v_1, v_2 \notin T \). Let \( P \) be a \( T \)-path from \( t_1 \) to \( t_2 \) which contains \( v_1 \). We know that such a path exists since \( v_1 \) lies in a main component. Let \( P_1 \) be the subpath of \( P \) from \( t_1 \) to \( v_1 \) and let \( P_2 \) be the subpath of \( P \) from \( v_1 \) to \( t_2 \). Let \( P_3 \) be a path from \( v_2 \) to \( t_3 \in T \) (\( t_3 \) can be the same as \( t_1 \) or \( t_2 \)). We know that such a path exists since \( v_2 \) is in a main component. We now consider the following two cases.

(a) \( P_3 \) does not intersect \( P_2 \) or \( P_1 \). Then clearly, there are paths from \( t_1 \) to \( t_3 \) and \( t_2 \) to \( t_3 \) which intersect the semi-isolated component \( C' \), and are disjoint from the solution, which is a contradiction since no vertex in \( C' \) can lie on a \( T \)-path disjoint from the solution.

(b) \( P_3 \) intersects \( P_2 \) or \( P_1 \). Without loss of generality, suppose that \( P_3 \) intersects \( P_2 \) first (see Fig. 7.2) when traversing from \( v_2 \) to \( t_3 \) and let the vertex at which this intersection occurs be \( u \). Let \( P'_3 \) be the subpath of \( P_3 \) from \( v_2 \) to \( u \) and let \( P'_2 \) be the subpath of \( P_2 \) from \( u \) to \( t_2 \). Additionally, let \( P \) be a path from \( v_1 \) to \( v_2 \) such that the internal vertices of \( P \) lie in \( C' \). Since \( C' \) is a connected component, we know that such a path exists. But now, \( P_1 + P + P'_3 + P'_2 \) is a \( t_1 \)-\( t_2 \) path disjoint from \( S \) and intersecting \( C' \). This is a contradiction since no vertex in \( C' \) can lie on a \( T \)-path in \( G \setminus S \) (by Observation 7.3.5).

This concludes the proof of the lemma.

\[ \square \]

**Definition 7.3.7.** Let \( (G, T = T_e \cup T_o, k) \) be an instance of PMWC and let \( S \) be a solution for this instance. Let \( C \) be a non-isolated component of \( G \setminus S \) and let \( C' \) be a semi-isolated component of \( C \). We denote the unique neighbor of \( C' \) in the adjacent main component by \( \chi(C') \) and refer to this vertex as the **pivot** of \( C' \). For every vertex \( v \in C' \), we define \( \chi(v) = \chi(C') \).
Lemma 7.3.8. Let \((G, T = T_e \cup T_o, k)\) be an instance of PMWC and let \(S\) be a solution for this instance. Then, no vertex in \(T\) occurs as a pivot in \(G \setminus S\).

Proof. Suppose that a terminal \(t \in T\) occurs as a pivot and let \(C\) be the component of \(G \setminus S\) containing \(t\). Recall that by assumption, every terminal in this instance has a single neighbor. Let \(u\) be the neighbor of \(t\) in \(G\). Therefore, it must be case that \(u\) lies in a semi-isolated component of \(C\). However, if \(C\) contains 2 terminals, then the main component of \(C\) is a connected subgraph containing both terminals (by Observation 7.3.5), which implies that the vertex \(u\) is also in the main component of \(C\), a contradiction. Therefore, it must be the case that \(C\) does not contain a terminal other than \(t\). But in this case, \(u\) is in the main component by definition, a contradiction. This completes the proof of the lemma.

7.3.2 Branching on isolated and semi-isolated vertices

In this subsection, we show that given an an isolated or semi-isolated vertex (Lemma 7.3.10 and Lemma 7.3.12) with respect to some solution, we can compute a set of bounded size which intersects some solution for this instance.

Lemma 7.3.9. Consider an instance \((G, T = T_e \cup T_o, k)\) of PMWC. Let \(S\) be a solution for this instance and let \(v\) be an isolated vertex in \(G \setminus S\). There is a solution for this instance which intersects an important \(v\)-\(T\) separator of size at most \(k\).

Proof. Due to Preprocessing Rule 2, \(v\) lies on a \(T\)-path in \(G\). Since \(v\) is isolated in \(G \setminus S\), it must be case that there is a non empty set \(K \subseteq S\) such that \(K\) is a minimal \(v\)-\(T\) separator. If \(K\) is an important \(v\)-\(T\) separator, then \(S\) itself a solution of the required kind. Suppose that \(K\) is not an important \(v\)-\(T\) separator and let \(J\) be an important \(v\)-\(T\) separator dominating \(K\). We claim that the set \(S' = (S \setminus K) \cup J\) is also a solution for
Input: Instance \((G, T = T_e \cup T_o, k)\) of PMWC and a vertex \(v\) such that \(v\) is isolated with respect to some solution for the instance

Output: A set \(R\) of vertices which intersects some solution

1. \(X \leftarrow \) set of important \(v\)-\(T\) separators of size at most \(k\)
2. \(R \leftarrow \bigcup_{X \in X} X\)
3. return \(R\)

Algorithm 7.3.1: Algorithm BRANCH-ISO to compute a set of vertices intersecting a solution when given a vertex isolated with respect to some solution.

the given instance. Clearly \(|S'| \leq |S|\). Therefore, if \(S'\) were not a solution, then there is a \(T\)-path of forbidden parity in the graph \(G \setminus S'\). Since \(S\) is a pmwc, this path must intersect \(S \setminus S' = K \setminus J\). This implies the existence of a path from \(K \setminus J\) to \(T\) in the graph \(G \setminus S'\) which in turn implies the existence of a path from \(K \setminus J\) to \(T\) in the graph \(G \setminus J\) since \(J \subseteq S'\).

However, since \(J\) dominates \(K\) (by Lemma 3.2.9), there is no path from \(K \setminus J\) to \(T\) in \(G \setminus J\), a contradiction. Therefore, \(S'\) is a solution for the given instance intersecting an important \(v\)-\(T\) separator of size at most \(k\). This completes the proof of the lemma.

**Lemma 7.3.10.** Consider an instance \((G, T = T_e \cup T_o, k)\) of PMWC. Let \(S\) be a solution for this instance and let \(v\) be an isolated vertex in \(G \setminus S\). Given \(v\), in time \(O^* (4^k)\), we can find a set of at most \(4^k k\) vertices with a non-empty intersection with some solution for this instance.

**Proof.** By Lemma 3.2.18 we know that the number of important \(v\)-\(T\) separators of size at most \(k\) is at most \(4^k\) and these can be enumerated in time \(O^* (4^k)\). Therefore, to compute the required set intersecting a solution, we simply enumerate all important \(v\)-\(T\) separators of size at most \(k\) and return the set obtained by taking the union of the vertices in these separators. By Lemma 7.3.9, the returned set intersects some solution for the given instance. This completes the proof of the lemma.
We now prove an analogous lemma for semi-isolated vertices.

**Lemma 7.3.11.** Consider an instance \((G, T = T_e \cup T_o, k)\) of PMWC. Let \(S\) be a solution for this instance and let \(v\) be a semi-isolated vertex in \(G \setminus S\). There is a solution intersecting an important \(v\)-\(T\) separator of size at most \(k + 1\) in the graph \(G\).

**Proof.** Due to Preprocessing Rule 2, \(v\) lies on a \(T\)-path in \(G\). Since \(v\) is semi-isolated, it must be the case that there is a non-empty set \(K \subseteq S\) such that \(K\) is a minimal \(v\)-\(T\) separator in the graph \(G \setminus \{\chi(v)\}\). If \(K' = K \cup \{\chi(v)\}\) is an important \(v\)-\(T\) separator, then \(S\) itself is a solution of the required kind. Suppose that \(K'\) is not an important \(v\)-\(T\) separator and let \(J\) be an important \(v\)-\(T\) separator dominating \(K'\). We now select a vertex \(u\) as follows. If \(\chi(v) \in J\), then we set \(u = \chi(v)\) and if \(\chi(v) \notin J\), then we choose as \(u\) an arbitrary vertex in \(J \setminus K\). We claim that the set \(S' = (S \setminus K) \cup (J \setminus \{u\})\) is also a solution for the given instance. Clearly, \(|S'| \leq |S|\). Therefore, if \(S'\) were not a solution, then there is a \(T\)-path intersecting \(S \setminus S' = K \setminus J\). This implies that there is a vertex \(z \in K \setminus J\) which has two vertex disjoint paths to \(T\) in \(G \setminus (J \setminus \{u\})\). However, since \(J\) dominates \(K'\), \(J\) intersects all paths from \(z\) to \(T\) in \(G\) (by Lemma 3.2.9). Therefore, there cannot be 2 vertex disjoint paths from \(z\) to \(T\) in the graph \(G \setminus (J \setminus \{u\})\), a contradiction. Hence, we conclude that \(S'\) is a solution for the given instance which intersects an important \(v\)-\(T\) separator of size at most \(k + 1\). This completes the proof of the lemma.

**Lemma 7.3.12.** Consider an instance \((G, T = T_e \cup T_o, k)\) of PMWC. Let \(S\) be a solution for this instance and let \(v\) be a semi-isolated vertex with respect to \(S\). Given \(v\), in time \(O^*(4^k)\), we can find a set of at most \(4^{k+1}(k + 1)\) vertices with a non-empty intersection with some solution for this instance.

**Proof.** By Lemma 3.2.18 we know that the number of important \(X\)-\(Y\) separators of size at most \(k + 1\) is at most \(4^{k+1}\) and these can be enumerated in time \(O^*(4^k)\). Therefore,
**Input**: Instance \((G, T = T_e \cup T_o, k)\) of PMWC and a vertex \(v\) such that \(v\) is semi-isolated with respect to some solution for the instance

**Output**: A set \(\mathcal{R}\) of vertices which intersects some solution

1. \(\mathcal{X} \leftarrow\) set of important \(v\)-\(T\) separators of size at most \(k + 1\)
2. \(\mathcal{R} \leftarrow \bigcup_{X \in \mathcal{X}} X\)
3. return \(\mathcal{R}\)

**Algorithm 7.3.2**: Algorithm BRANCH-SEMI-ISO to compute a set of vertices intersecting a solution when given a vertex semi-isolated with respect to some solution.

To compute the required set intersecting a solution, we simply enumerate all important \(v\)-\(T\) separators of size at most \(k + 1\) and return the set obtained by taking the union of the vertices in these separators (see Algorithm 7.3.2). By Lemma 7.3.11, the returned set intersects some solution for the given instance. This completes the proof of the lemma.

**Definition 7.3.13.** Consider an instance \((G = (V, E), T, k)\) of PMWC. We call \(J \subseteq V\) an important component if \(G[J]\) is connected and \(N(J)\) is an important \(J\)-\(T\) separator of size at most \(k + 1\).

**Lemma 7.3.14.** Let \((G, T = T_e \cup T_o, k)\) be a YES instance of PMWC. Then, there is a solution \(S\) for this instance such that every every isolated or semi-isolated component in \(G \setminus S\) is an important component.

**Proof.** Consider the set of solutions for this instance with minimum size and among these let \(S\) be the solution which maximizes the number of vertices which are isolated or semi-isolated in \(G \setminus S\). We claim that \(S\) is a solution which satisfies the statement of the lemma.

We first show that for any isolated or semi-isolated component \(C\) in \(G \setminus S\), the set \(N(C)\) is a minimal \(C\)-\(T\) separator. For this, it suffices to show that for any vertex
\( v \in N(C) \), there is a \( C-T \) path in the graph \( G \setminus (N(C) \setminus \{v\}) \). Consider a vertex \( v \in N(C) \). Suppose that \( C \) is a semi-isolated component and \( v = \chi(C) \). There is a path from \( v \) to \( T \) in \( G \setminus (N(C) \setminus \{v\}) \) since \( v \) lies in a main component of \( G \setminus S \) by definition and \( N(C) \setminus \{v\} \) is contained in \( S \). This implies the presence of a \( C-T \) path in \( G \setminus (N(C) \setminus \{v\}) \) since \( C \) is adjacent to \( v \). Therefore, we may assume that if \( C \) is semi-isolated, then \( v \neq \chi(C) \), which implies that \( v \in S \). Now, since the solution \( S \) is minimal, there is a \( T \)-path containing \( v \) in \( G \setminus (S \setminus \{v\}) \), which implies that there are 2 vertex disjoint paths from \( v \) to \( T \) in \( G \setminus (N(C) \setminus \{v\}) \) since \( C \) is adjacent to \( v \). Therefore, we conclude that \( N(C) \) is indeed a minimal \( C-T \) separator.

For any isolated or semi-isolated component \( C \), \( |N(C)| \leq k + 1 \). Therefore, it remains to show that the neighborhood of every isolated or semi-isolated component is an important separator.

**Case 1: isolated components.** Consider an isolated component \( C \) such that the set \( X = N(C) \) is not an important \( C-T \) separator. Let \( Y \) be an important \( C-T \) separator dominating \( X \). We claim that \( S' = (S \setminus X) \cup Y \) is also a solution for the instance. Clearly, \( |S'| \leq |S| \). Therefore, if \( S' \) were not a solution, then there is a \( T \)-path intersecting a vertex in \( S \setminus S' = X \setminus Y \) in the graph \( G \setminus S' \). This implies that there is a path from a vertex in \( X \setminus Y \) to \( T \) in the graph \( G \setminus Y \). But \( X \setminus Y \) is separated from \( T \) by \( Y \) (by Lemma 3.2.9) leading to a contradiction. Therefore, \( S' \) is indeed a solution for this instance. Observe that if \( |Y| < |X| \) or \( (Y \setminus X) \cap S \neq \emptyset \), then \( S' \) is strictly smaller than \( S \), which contradicts our choice of \( S \). Therefore, we may assume that \( |Y| = |X| \) and \( (Y \setminus X) \cap S = \emptyset \). Let \( y \) be an arbitrary vertex in \( Y \setminus X \). We claim that \( S'' = S' \setminus \{y\} \) is also a solution for this instance. If this were not the case, then there is a \( T \)-path in-
intersecting $y$ in $G \setminus S''$. Since $S$ is a solution, this path must intersect some vertex in $S \setminus S''$. Since $y \notin S$, it must be the case that this path intersects a vertex $z \in X \setminus Y$. Therefore, we have that there is a vertex $z \in X \setminus Y$ which has 2 vertex disjoint paths to $T$ in $G \setminus S''$ and hence $z$ has 2 vertex disjoint paths to $T$ in $G \setminus (Y \setminus \{y\})$. However, we know that $Y$ intersects all paths from $z$ to $T$ in $G$ (by Lemma 3.2.9). Therefore, there cannot be 2 vertex disjoint paths from $z$ to $T$ in $G \setminus (Y \setminus \{y\})$, a contradiction. Hence, we conclude that $S''$ is also a solution for this instance. But $S''$ is strictly smaller than $S'$, which in turn is no larger than $S$. Therefore, we have that $S''$ is a solution strictly smaller than $S$, which contradicts our choice of $S$. Hence, we conclude that $N(C)$ is indeed an important $C$-$T$ separator.

**Case 2: semi-isolated components.** Consider a semi-isolated component $C$ such that the set $X = N(C)$ is not an important $C$-$T$ separator. Let $Y$ be an important $C$-$T$ separator dominating $X$. We select a vertex $u$ as follows. If $\chi(C) \in Y$ then $u = \chi(C)$ and if $\chi(C) \notin Y$ then choose an arbitrary vertex in $Y \setminus X$ as $u$. We claim that $S' = (S \setminus X) \cup (Y \setminus \{u\})$ is also a solution for the instance. Clearly, $|S'| \leq |S|$. Therefore, if $S'$ were not a solution, then there is a $T$-path intersecting a vertex $z$ in $S \setminus S' = X \setminus Y$, implying that there are 2 vertex disjoint paths from $z$ to $T$ in $G \setminus (Y \setminus \{u\})$. But $X \setminus Y$ is separated from $T$ by $Y$ (by Lemma 3.2.9). Therefore, there cannot be 2 vertex disjoint paths from $z$ to $T$ in $G \setminus (Y \setminus \{u\})$, a contradiction. Therefore, $S'$ is indeed a solution for this instance. Observe that if $|Y| < |X|$ or $(Y \setminus (X \cup \{u\})) \cap S \neq \emptyset$, then $S'$ is strictly smaller than $S$, which contradicts our choice of $S$. Therefore, we may assume that $|Y| = |X|$ and $(Y \setminus (X \cup \{u\})) \cap S = \emptyset$.

We now prove the following properties regarding the set $Y$ defined above.
Claim 6. (a) Every vertex in $R(C,Y)$ is isolated or semi-isolated in $G \setminus S'$.
(b) For every vertex $a \in NR(C,Y) \cup \{u\}$, if $a$ is isolated or semi-isolated in $G \setminus S$, then $a$ is isolated or semi-isolated in $G \setminus S'$.

Proof. (a) Consider a vertex $x$ in $R(C,Y)$ which lies in a main component of $G \setminus S'$. Then, $x$ lies on a $T$-path in $G \setminus S'$ and hence there are 2 vertex disjoint paths from $x$ to $T$ in $G \setminus S'$, which implies that there are 2 vertex disjoint paths from $x$ to $T$ in $G \setminus (Y \setminus \{u\})$. However, $Y$ is an $x$-$T$ separator in $G$ by definition and hence there cannot be 2 vertex disjoint paths from $x$ to $T$ in $G \setminus (Y \setminus \{u\})$, a contradiction. Therefore, we conclude that every vertex in $R(C,Y)$ is either isolated or semi-isolated in $G \setminus S'$.

(b) Consider a vertex $a \in NR(C,Y) \cup \{u\}$ such that $a$ is isolated or semi-isolated in $G \setminus S$, but lies in a main component of $G \setminus S'$. Since $a$ does not have 2 vertex disjoint paths to $T$ in $G \setminus S$, it must be the case that at least one of the two vertex disjoint paths from $a$ to $T$ in $G \setminus S'$ intersects the set $S \setminus S'$. This implies that there is a vertex $z \in S \setminus S'$ which has 2 vertex disjoint paths to $T$ in $G \setminus S'$. However, this implies that $z \in X \setminus Y$, which is contained in $R(C,Y)$ by definition and we have already established that every vertex in $R(C,Y)$ is isolated or semi-isolated in $G \setminus S'$ (see (a)), which is a contradiction. This completes the proof of the claim. 

We now compare the number of isolated and semi-isolated vertices in $G \setminus S$ against those in $G \setminus S'$. The number of isolated and semi-isolated vertices from the set $NR(C,Y) \cup \{u\}$ in $G \setminus S'$ is at least that in $G \setminus S$ (Claim 6 (b)). Similarly, the number of isolated and semi-isolated vertices from the set $R(C,Y) \setminus X$ in $G \setminus S'$ is also at least that in $G \setminus S$ (Claim 6 (a)). Therefore, it remains to compare the number of isolated and semi-isolated vertices in $G \setminus S$ which lie in $Y \setminus (X \cup \{u\})$ and the number of isolated and semi-isolated vertices in $G \setminus S'$ which lie in $X \setminus Y$. Again, by Claim 6 (a), the number
of isolated or semi-isolated vertices in $G \setminus S'$ which lie in $X \setminus Y$ is exactly $|X \setminus Y|$. The number of isolated or semi-isolated vertices in $G \setminus S$ which lie in $Y \setminus (X \cup \{u\})$ is at most $|Y \setminus X| - 1$ since $u$ is disjoint from $X \cap Y$. Therefore, the number of isolated or semi-isolated vertices in $G \setminus S'$ is at least $|X \setminus Y| - (|Y \setminus X| - 1)$ more than that in $G \setminus S$. Since by our assumption, $|Y| = |X|$, we have that $|X \setminus Y| = |Y \setminus X|$. Therefore, we have a solution $S'$ with strictly more isolated or semi-isolated vertices, which contradicts our choice of $S$. This completes the proof of the lemma.

**Lemma 7.3.15.** Every vertex $v \in V$ is contained in at most $4^{k+1}$ important components. Furthermore, all important components in $G$ can be enumerated in time $O^*(4^k)$.

**Proof.** Consider an important component $J$. Then, $N(J)$ is an important $v$-$T$ separator for any $v \in J$ (Lemma 3.2.17(1)). Since there are at most $4^{k+1}$ important $v$-$T$ separators of size $k + 1$, $v$ can occur in at most $4^{k+1}$ important components. The set of important components can be computed by computing the set of important $v$-$T$ separators for every vertex $v$ in the graph. By Lemma 3.2.18, this requires time $O^*(4^k)$.

**Lemma 7.3.16.** Let $G = (V, E)$ be a graph and $T$ be a vertex set such that every vertex in $G$ lies on a $T$-path and every $T$-path in $G$ has the same parity. Then $G$ is a bipartite graph.

**Proof.** Suppose that this is not the case and let $C$ be an odd cycle in $G$. Consider a vertex $w$ in the cycle which lies on a $t_1$-$t_2$ path $P$ where $t_1, t_2 \in T$. Clearly $t_1$ and $t_2$ cannot both lie on $C$, since it implies the presence of an odd $T$-path. Hence, at least one of the vertices, say $t_1$, is disjoint from $C$. Now, when traversing $P$ from $t_1$ to $t_2$, let $u$ be the first vertex of $P$ which is in $C$ and let $v$ be the last vertex of $P$ which is in $C$. Note that $v$ and $t_2$ need not be distinct. Since $P$ intersects $w$, it intersects $C$ and hence the vertex $u$ exists. Now, let $P_1$ be the subpath of $P$ from $t_1$ to $u$ and $P_2$ be the subpath of
Suppose that $u$ and $v$ are distinct and let $P_o$ and $P_e$ be the 2 subpaths of $C$ between $u$ and $v$ where $P_o$ is odd and $P_e$ is even. Then, the two paths $P_1 + P_o + P_2$ and $P_1 + P_e + P_2$ are $T$-paths with different parities, which is a contradiction. Hence, it must be the case that $u = v$ and that any $T$-path intersecting $w$ intersects $C$ only in $w$. Now, let $u'$ be a vertex in $C$ distinct from $w$ and let $P'$ be a $t_3$-$t_4$ path on which $w'$ lies, where $t_3, t_4 \in T$. Note that $t_3$ and $t_4$ need not be distinct from $t_1$ or $t_2$. Let $P'_1$ be the subpath of $P'$ from $t_3$ to $w'$ and let $P'_2$ be the subpath of $P'$ from $w'$ to $t_4$. By our earlier arguments, it must be the case that $P'_1$ and $P'_2$ intersect $C$ only in the vertex $w'$. Now, if the paths $P_1, P_2, P'_1$ and $P'_2$ are all pairwise disjoint, then we have a $T$-path passing containing $w$ which intersects $C$ in at least 2 vertices, a contradiction to our earlier assumption that any $T$-path containing $w$ does not intersect $C$ in any other vertex. Hence, we consider the case when $P_1$ or $P_2$ intersects $P$. We assume without loss of generality that $P'_1$ intersects $P$. While traversing $P'_1$ from $w'$ to $t_3$, let $x$ be the first vertex of $P$ on $P'_1$ and suppose that $x \in P_1$. Let the subpath of $P'_1$ from $w'$ to $x$ be $P'_1$ and the subpath of $P_1$ from $t_1$ to $x$ be $\tilde{P}_1$. Then, the paths $\tilde{P}_1 + P'_1$ and $P_2$ are disjoint paths from distinct vertices of $T$ to distinct vertices of $C$. This is a contradiction to our assumption that any $T$-path intersects $C$ in at most 1 vertex. This completes the proof of the lemma.

**Observation 7.3.17.** Let $(G, T = T_e \cup T_o, k)$ be an instance of PMWC and let $S$ be a solution for this instance. The main components of $G \setminus S$ are bipartite and any odd cycle in $G \setminus S$ has at most 1 vertex in a main component.

**Proof.** Fix a main component in $G \setminus S$ and suppose that this component contains an odd cycle. This implies that the main component contains at least 3 vertices and therefore, there are at least 2 terminals in this main component. By Observation 7.2.2, the terminals in this main component are either all even terminals or all odd terminals. Therefore, the
$T$-paths which lie in this main component are all even or all odd. By Lemma 7.3.16, the main component is bipartite.

Suppose that there is an odd cycle in $G \setminus S$ which has two vertices in a main component. Since we have already shown that the main components are bipartite, this cycle must intersect a semi-isolated component as well. This implies that there is a semi-isolated vertex $v$ which has 2 vertex disjoint paths to 2 distinct vertices in the adjacent main component, which is a contradiction to Lemma 7.3.6.

Lemma 7.3.18. Let $(G, T = T_e \cup T_o, k)$ be an instance of PMWC and let $S$ be a solution for this instance of the kind described in Lemma 7.3.14 and suppose that $G \setminus S$ is non-bipartite. Then, there is an important component $C$ such that $C$ is disjoint from $S$ and $G[C \cup N(C)]$ is non-bipartite.

Proof. Consider an odd cycle in $G \setminus S$. By Observation 7.3.17, at most one vertex of this cycle lies in a main component. Therefore, there is an isolated or semi-isolated component $C$ such that this odd cycle lies in the graph $G[C \cup N(C)]$. However, by Lemma 7.3.14, every isolated and semi-isolated component in $G \setminus S$ is an important component and therefore, $C$ is an important component disjoint from $S$ such that $G[C \cup N(C)]$ is non-bipartite. This completes the proof of the lemma.

We refer to any set which intersects all odd cycles in a graph as an oct of the graph. We now describe the procedure to obtain an instance of PMWC with a solution that is also an oct for the input graph. Formally, we have the following lemma.

Lemma 7.3.19. Let $(G, T = T_e \cup T_o, k)$ be an instance of PMWC. If $(G, T = T_e \cup T_o, k)$ is a YES instance, then there is a solution for this instance which is also an oct of $G$ or there is a set $Z$ of $2^{O(k)}$ vertices which can be computed in time $O^*(2^{O(k)})$ such that $Z$ intersects some solution for this instance.
Proof. Suppose that \((G, T = T_e \cup T_o, k)\) is a YES instance and there is no solution for this instance which is also an oct of \(G\). Let \(S\) be a solution for this instance such that every isolated or semi-isolated component is an important component. By Lemma 7.3.14, we know that such a solution exists. By our assumption, there is an odd cycle in \(G \setminus S\).

Let \(\mathcal{H}\) be the set of all important components in \(G\). Let \(\mathcal{J}\) be the set of all components \(H \in \mathcal{H}\) such that \(G[H \cup N(H)]\) is non-bipartite. Let \(\mathcal{M}\) be an arbitrary subset of \(\mathcal{J}\) of size \(4^{k+1}k + 1\). If \(|\mathcal{J}| < 4^{k+1}k + 1\), then set \(\mathcal{M} = \mathcal{J}\). Let \(M_1, \ldots, M_\ell\) be the components in \(\mathcal{M}\). We now define the set \(Z\) as follows. Initially set \(Z = \bigcup_{i=1}^\ell N(M_i)\). For each \(M_i\), select two vertices \(u_i\) and \(v_i\) in an arbitrary odd cycle in \(G[M_i \cup N(M_i)]\). For each \(1 \leq i \leq \ell\), invoke the algorithms of Lemma 7.3.10 (Algorithm 7.3.1) and Lemma 7.3.12 (Algorithm 7.3.2) on the vertices \(u_i\) and \(v_i\) and add the vertices returned by each of these invocations to \(Z\). Finally, return \(Z\). We claim that \(Z\) intersects some solution for this instance. Suppose that this is not the case.

We first show that for some \(M \in \mathcal{M}\), any odd cycle in \(G[M \cup N(M)]\) is also present in \(G \setminus S\).

We first consider the case when \(|\mathcal{M}| = 4^{k+1}k + 1\). Since at most \(4^{k+1}k\) important components can intersect \(S\), there is an \(M \in \mathcal{M}\) such that \(M\) is disjoint from \(S\). Furthermore, by definition \(N(M) \subseteq Z\) and by our assumption, \(Z\) is disjoint from \(S\). Therefore, any odd cycle in the graph \(G[M \cup N(M)]\) is an odd cycle in \(G \setminus S\).

We now consider the case when \(|\mathcal{M}| \leq 4^{k+1}k\). Recall that in this case, \(\mathcal{M} = \mathcal{J}\). By Lemma 7.3.18, there is at least one important component \(M\) such that \(G[M \cup N(M)]\) is non-bipartite. Since, every important component in \(G\) with the property that \(G[C \cup N(C)]\) is non-bipartite is present in \(\mathcal{J}\), \(M\) is also present in \(\mathcal{M}\). Since by definition \(N(M) \subseteq Z\) and by our assumption, \(Z\) is disjoint from \(S\), we have that \(N(M)\) is disjoint from \(S\). Therefore, any odd cycle in the graph \(G[M \cup N(M)]\) is an odd cycle in \(G \setminus S\).
Without loss of generality, we assume that \( M = M_1 \in \mathcal{M} \).

Recall that since we have assumed that \( Z \) is disjoint from \( S \) and \( Z \) contains \( N(M_1) \) by definition, \( S \) is also disjoint from \( N(M_1) \). Consider the vertices \( u_1 \) and \( v_1 \) selected by the algorithm from an odd cycle in \( G[M \cup N(M)] \). By Observation 7.3.17, at least one of these two vertices is either isolated or semi-isolated in \( G \setminus S \) and by Lemma 7.3.10 and Lemma 7.3.12, the sets returned by Algorithm 7.3.1 or Algorithm 7.3.2 indeed intersect some solution for the given instance. However, this leads to a contradiction since \( Z \) contains the vertices returned by these two invocations. Therefore, we conclude that \( Z \) intersects some solution for the given instance. It remains to prove the bound on the size of \( Z \) and the time required to compute \( Z \).

The size of \( Z \) is bounded by \(|\mathcal{M}|(k + 1) + 2 \cdot 2^{O(k)} = 2^{O(k)}\). The time required to compute \( Z \) is bounded by the time required to compute the set of important components in \( G \) and the time required to run Algorithm 7.3.1 and Algorithm 7.3.2 \( 2|\mathcal{M}| \) times. Therefore, \( Z \) can be computed in time \( O^*(2^{O(k)}) \). This completes the proof of the lemma.

Lemma 7.3.20. Given an instance \((G, T = T_e \cup T_o, k)\) of PMWC, there is an algorithm which runs in time \( O^*(2^{O(k^2)}) \) and returns \( \ell \) instances \( \{(G_i, T_i, k_i)\}_{1 \leq i \leq \ell} \) of PMWC, where \( \ell = 2^{O(k^2)} \) such that

1. If \((G, T = T_e \cup T_o, k)\) is a YES instance, then for some \( 1 \leq i \leq \ell \), the instance \((G_i, T_i, k_i)\) is a YES instance of PMWC and has a solution which is also an oct of \( G_i \).

2. If \((G, T = T_e \cup T_o, k)\) is a NO instance, then all the returned instances are NO instances of PMWC.

Proof. The algorithm simply applies Lemma 7.3.19, constructs the set \( Z \) and branches...
on each vertex in this set. There is also an additional branch in which the current instance is returned. The correctness of this algorithm follows from Lemma 7.3.19. We now prove the stated bound on running time and the number of returned instances.

Since $|Z| = 2^{O(k)}$, at every step, the algorithm branches in $2^{O(k)}$ ways. Out of these branches, one is a leaf (where the current instance is returned) and in every other branch, $k$ decreases by 1 since we are adding a vertex to the solution. Hence, the number of leaves in the recursion tree is bounded by $2^{O(k^2)}$. The time taken at each step is the time required to compute $Z$, which is $O^*(2^{O(k)})$ by Lemma 7.3.19. Therefore, the algorithm runs in time $O^*(2^{O(k^2)})$ and returns $2^{O(k^2)}$ instances. This completes the proof of the lemma.

We apply the Lemma 7.3.20 each of the instances returned by the algorithm of Lemma 7.2.4 and therefore, we may assume that from this point on, we are dealing with the following problem.

**BIPARTITION PARITY MULTIWAY CUT (BPMWC)**

**Parameter:** $k$

**Input:** An instance $(G, T = T_e \cup T_o, k)$ of PMWC where $|T_e| \leq 6k$, with the guarantee that if this instance is a YES instance, then there is a solution which is also an oct of $G$.

**Question:** Is there a pmwc of size at most $k$ disjoint from $T$?

Before we move towards applying the important separator template to solve this problem, we perform one final preprocessing on the instance.

**Lemma 7.3.21.** There is an algorithm that, given an instance $I = (G, T = T_e \cup T_o, k)$ of BPMWC, runs in time $O^*(2^{O(k \log k)})$ and returns $2^{k \log k}$ instances of BPMWC $\{(G_i, T = T_e \cup T_o, k)\}_{1 \leq i \leq \ell}$ where $\ell = 2^{k \log k}$ such that
1. If \((G, T = T_e \cup T_o, k)\) is a \textsc{Yes} instance of BPMWC then there is an \(1 \leq i \leq \ell\) such that \(I_i = (G_i, T = T_e \cup T_o, k)\) is a \textsc{Yes} instance of BPMWC and has an oct solution \(S_i\) such that if an even terminal is non-adjacent to the rest of the terminals, then it occurs by itself in a connected component of \(G_i \setminus S_i\).

2. If \((G, T = T_e \cup T_o, k)\) is a \textsc{No} instance \(I_i = (G_i, T = T_e \cup T_o, k)\) is a \textsc{No} instance of BPMWC for every \(1 \leq i \leq \ell\).

\textbf{Proof.} Let \(\mathcal{P}\) be the set of all partitions of the set \(T_e\) with the property that no set in any partition contains more than 2 vertices of \(T_e\). Since \(T_e \leq 6k\), there are \(2^{O(k \log k)}\) such partitions. Let \(\mathcal{P} = \{P_1, \ldots, P_\ell\}\). We construct \(\ell\) instances of BPMWC each corresponding to a partition in \(\mathcal{P}\) as follows. Consider a partition \(P_i = (X_1, \ldots, X_r)\) where each \(X_j\) contains at most 2 vertices of \(T_e\). Let \(G_i\) be the graph obtained from \(G\) by adding an edge between every pair of even terminals which occur in the same set in the partition \(P_i\). Finally, the instances \(\{(G_i, T = T_e \cup T_o, k)\}_{1 \leq i \leq \ell}\) are returned. The bound on the running time of the algorithm is clear.

1. Suppose that \(I\) is a \textsc{Yes} instance and let \(S\) be a solution for this instance such that \(S\) is also an oct for \(G\). Let \(P = (X_1, \ldots, X_r)\) be the partition of \(T_e\) among the connected components of \(G \setminus S\), that is, the vertices in each \(X_i\) occur in the same connected component of \(G \setminus S\) and the vertices of any \(X_i\) and \(X_j\) occur in different components of \(G \setminus S\) for \(i \neq j\). Without loss of generality, let \(P_1\) be this partition. We claim that the instance \(I' = (G_1, T = T_e \cup T_o, k)\) is a \textsc{Yes} instance of BPMWC. In order to prove this, we show that \(S\) is also a solution for the instance \(I'\) and \(S\) is an oct for the graph \(G_1\).

If \(S\) were not a solution for the instance, then there is a \(T\)-path \(P\) of forbidden parity in \(G_1 \setminus S\). Since \(S\) is a solution for \(G \setminus S\), it must be the case that \(P\) contains an edge
which is in $G_1$ but not in $G$. Therefore, $P$ contains an edge between two even terminals $t_1$ and $t_2$. However, by assumption, $t_1$ and $t_2$ occur in the same component of $G \setminus S$ and there is no other terminal in this component. Since we only added edges between vertices in the same component of $G \setminus S$, $t_1$ and $t_2$ appear in the same connected component of $G_1 \setminus S$ and no other terminals occur in this component. Therefore, it must be the case that $P$ is a path between $t_1$ and $t_2$. Since we have already established that $P$ contains an edge between $t_1$ and $t_2$, $P$ itself must be just the edge $(t_1, t_2)$. But by assumption, $P$ is a path of even parity, which contradicts our conclusion that $P$ has length 1. Clearly, any even terminal in $G_1$ which is non-adjacent to the rest of the terminals occurs by itself in a connected component of $G \setminus S$ and hence occurs by itself in a connected component of $G_1 \setminus S$.

2. In the converse direction, since adding edges to the graph $G$ cannot create a smaller pmwc, if $I$ were a No instance then $(G_i, T = T_e \cup T_o, k)$ will be a No instance for every $1 \leq i \leq \ell$.

\[\square\]

We assume that the instance of BPMWC we are given is one returned by the algorithm of Lemma 7.3.21. Furthermore, observe that in order to prove Theorem 7.1.1, it suffices to prove the following lemma.

**Lemma 7.3.22.** BPMWC can be solved in time $O^*(2^{O(k^3)})$.

In the rest of the chapter, we describe how to apply the important separator template to prove this lemma.
7.4 Phase 1 of the important separator template

The following lemma gives an algorithm for the case when the set of even terminals in the given instance is empty.

Lemma 7.4.1. Let \((G, T = T_e \cup T_o, k)\) be an instance of BPMWC where \(T_e\) is empty. Then, we can solve this problem in time \(O^*(2.3146^k)\).

Proof. The proof is by a parameter preserving reduction to the variable version of ALMOST 2-SAT, called the ALMOST 2 SAT\((\nu)\) problem, which can then be solved in \(O^*(2.3146^k)\) time (Corollary 6.5.2). The reduction is as follows. For every vertex \(u\) of the graph, we have a variable \(x_u\). The variable \(x_u\) is intended to represent the side of the (fixed) bipartition of \(G \setminus S\) which contains \(u\). The 2-CNF formula is constructed as follows. For every edge \((t, u)\) where \(t \in T\), add a clause \((u)\). For every edge \((u, v)\) in the graph, add two clauses \((x_u \lor x_v)\) and \((\bar{x}_u \lor \bar{x}_v)\) to the 2-CNF formula \(F\). This completes the construction of \(F\). We remark that both these clauses will be satisfied if and only if \(x_u\) and \(x_v\) are assigned different values. In addition, we also remark that the subformula \(F_P\) of \(F\) induced by the clauses corresponding to the edges of some odd \(T\)-path \(P\) is unsatisfiable, that is, we cannot find a satisfying assignment for it unless we delete at least 1 variable from this subformula. We claim that if \((G, T = T_e \cup T_o, k)\) is a YES instance of BPMWC, then \((F, k)\) is a YES instance of ALMOST 2 SAT\((\nu)\) and if \((F, k)\) is a YES instance of ALMOST 2 SAT\((\nu)\), then \((G, T, k)\) is also a YES instance of BPMWC.

Suppose \((G, T, k)\) has a solution \(S\) such that \(G \setminus S\) is bipartite. Let \(S_v\) be the set of the variables corresponding to the vertices in \(S\). We claim that the formula \(F' = F \setminus S_v\) is satisfiable. Consider the following assignment for \(F'\). Fix a bipartition for \(G \setminus S\) such that all the vertices in \(T_o\) occur in the same partition. Such a partition is possible
since for each connected component of $G \setminus S$, the vertices in $T_o$ occur in the same partition of any bipartition. For every vertex $u$ which lies in the same partition as $T_o$ in the fixed partition of $G \setminus S$, we assign $x_u = 1$ and for all other vertices, we assign the corresponding variable the value 0. This assignment clearly satisfies $F'$.

Conversely, consider a solution $S_e$ for the instance $(F, k)$, let $F' = F \setminus S_e$, and let $S$ be the set of vertices corresponding to the variables in $S_e$. If a terminal occurs in $S$, then we replace it with its neighboring vertex (recall that every odd terminal is adjacent to a single vertex). We claim that $S$ is a solution for the given instance of BPMWC. Suppose that this is not the case, and consider an odd $T$-path $P$ in the graph $G \setminus S$ and let $F_P$ be the subformula of $F$ induced by the clauses corresponding to the edges in $P$. Since none of the vertices intersecting the path $P$ have been deleted, it must be the case that none of the variables corresponding to the vertices along this path are in $S_e$. But this implies that $F_P$ remains as a subformula of $F'$ and since $F_P$ is not satisfiable, $F'$ is also not satisfiable, which is a contradiction. \[\square\]

**Lemma 7.4.2.** Let $(G, T = T_e \cup T_o, k)$ be an instance of BPMWC where $T_o$ is empty and $|T_e| = 2$. Then, we can solve this problem in time $O^*(2.3146^k)$.

**Proof.** Let $T_e = \{t_1, t_2\}$. Let $G'$ be the graph obtained from $G$ by subdividing the edges incident on $t_1$, let $T'_e = \emptyset$ and let $T'_o = T_e$. It is easy to see that $(G, T = T'_e \cup T'_o, k)$ is a YES instance of BPMWC if and only if $(G', T' = T'_e \cup T'_o, k)$ is a YES instance of BPMWC. Therefore, by Lemma 7.4.1, the problem can be solved in time $O^*(2.3146^k)$. \[\square\]

Finally, the following lemma is a straightforward consequence of Lemma 7.3.21.

**Lemma 7.4.3.** Let $(G, T = T_e \cup T_o, k)$ be an instance of BPMWC where $|T_e| \geq 3$ or $T_e$ and $T_o$ are both non-empty. Let $T_1 \subseteq T_e$ such that either $T_1$ consists of a single even
terminal non-adjacent to the rest of the terminals or it consists of a pair of adjacent even
terminals. Then, there is an oct solution for the given instance containing a minimal $T_1$-
$(T \setminus T_1)$ separator.

We have therefore, shown that either the problem can be solved straightaway in FPT
time by applying Lemma 7.4.1 or Lemma 7.4.2 or there is a solution which contains
a minimal $T_1$-$ (T \setminus T_1)$ separator. This completes Phase 1 of the important separator
template.

7.5 Phases 2 and 3

In this section, we describe the second and third phases of the template.

7.5.1 Tight separator sequences and a generalization of important
separators

In this subsection we define the notion of a tight separator sequence and use it in the
context of BPMWC to define a generalization of important separators with the proper-
ties we require. This serves as the dominating set described in the important separator
template.

**Definition 7.5.1.** Let $G = (V, E)$ be a graph and let $X, Y \subseteq V$ be disjoint vertex
sets. We define an important $X$-$Y$ separator of order $i$, $S^i$ to be the unique smallest
important $X$-$S^{i-1}$ separator in $G$, where $S^0 = Y$.

**Definition 7.5.2.** Let $G = (V, E)$ be a graph and let $X, Y \subseteq V$. Let $l \geq 1$ be such that
there is no important $X$-$Y$ separator of order $l + 1$. We define a tight $X$-$Y$ separator
sequence $I$ to be a set $I = \{S^i | 1 \leq i \leq l\}$, where $S^i$ is an important $X$-$Y$ separator of
order $i$. (see Fig. 7.3).
**Input**: \((G, T_1, T_2, k)\)

**Output**: Tight sequence of \(T_1\)-\(T_2\) separators if there is a \(T_1\)-\(T_2\) separator of size at most \(k\) and \(\text{No otherwise}\)

1. Apply Lemma 3.2.15 to compute the unique smallest important \(T_1\)-\(T_2\) separator \(Y\)
2. \(I = Y\)
3. if \(|Y| > k\) then return \(\text{No}\).
4. while there is an important \(T_1\)-\(Y\) separator \(X\) of size at most \(|Y|\) do
5. \(I = I \cup \{X\}\)
6. \(Y = X\)
7. end
8. return \(I\)

**Algorithm 7.5.1**: Algorithm COMPUTE-TIGHT-SEQUENCE to compute the Tight Sequence of \(T_1\)-\(T_2\) separators.

**Observation 7.5.3.** Let \(G = (V, E)\) be a graph and let \(X, Y \subseteq V\). Given two \(X-Y\) separators \(S_1\) and \(S_2\), we say that \(S_1 \preceq S_2\) if \(S_2\) covers \(S_1\) with respect to \(X\). Then, \((I, \preceq)\) forms a total order where \(I\) is a tight \(X-Y\) separator sequence.

Observation 7.5.3 is the reason we refer to the set \(I\) as a sequence.

**Lemma 7.5.4.** Let \(G = (V, E)\) be a graph and let \(X, Y \subseteq V\). A tight \(X-Y\) separator sequence is unique and can be computed in polynomial time.

**Proof.** We know that for any two disjoint vertex subsets \(A\) and \(B\), there is a unique smallest important \(A-B\) separator (by Lemma 3.2.13) and it can be computed in polynomial time. Therefore, for every \(i\), an important \(X-Y\) separator of order \(i\) is unique and can be computed in polynomial time. The algorithm to compute the tight \(X-Y\) separator sequence is described in Algorithm 7.5.1.

**Lemma 7.5.5.** Let \(G = (V, E)\) be a graph, let \(X, Y \subseteq V\) and let \(I\) be the tight \(X-Y\) separator sequence. Let \(P_1\) and \(P_2\) be two separators in \(I\) such that \(P_1 \preceq P_2\) and there is no \(P_3\) in \(I\) such that \(P_1 \preceq P_3 \preceq P_2\). Then, the size of a minimum \(X-Y\) separator which lies in the set \(NR(X, P_1) \cap R(X, P_2)\) is at least \(|P_1| + 1\).
Proof. If this were not the case, then there is an $X$-$Y$ separator $S$ of size $|P_1|$ which lies in the set $NR(X, P_1) \cap R(X, P_2)$. Since $P_2$ is a successor of $P_1$ in $I$, it must be the case that $S \notin I$. Let $i$ be such that $P_2 = S^{i-1} = S^i$, that is, $P_1$ is the unique smallest important $X$-$P_2$ separator. However, since $P_1 \preceq S \succeq P_2$ and $S$ lies in the set $NR(X, P_1) \cap R(X, P_2)$ it implies that $S$ is an $X$-$P_2$ separator dominating $P_1$, which contradicts the fact that $P_1$ is an important $X$-$P_2$ separator.

The key consequence of the definition of the tight separator sequence is that it defines a natural partition of the graph into slices with small boundaries. Using this, we may restrict our search to local parts of the graph, in which case finding separators with certain properties becomes easier. We will now describe how this concept is applied in the context the BPMWC problem.

**Definition 7.5.6.** For a graph $G = (V, E)$, we refer to a smallest odd cycle transversal in the graph $G$ as a minimum oct and denote the size of this set by $\text{oct}(G)$.

**Definition 7.5.7.** Let $G = (V, E)$ be a graph, let $X, Y \subseteq V$. Let $S$ be a minimal $X$-$Y$ separator such that $\text{oct}(G[R(X, S)]) = \ell$. We say that a minimal $X$-$Y$ separator $S'$ well dominates $S$ (with respect to $X$) if $S'$ dominates $S$ with respect to $X$ and $\text{oct}(G[R(X, S')]) \leq \ell$.

Note that any $X$-$Y$ separator well dominates itself. The above definition is motivated by the following observation.

**Observation 7.5.8.** Let $(G, T = T_e \cup T_o, k)$ be an instance of BPMWC and let $T_1 \subseteq T_e$ such that $T_1 = \{t\}$ where $t$ is an even terminal non-adjacent to any other terminal or $T_1 = \{t_1, t_2\}$ where $t_1$ and $t_2$ are adjacent in $G$. Let $S$ be a solution for this instance such that $G \setminus S$ is a bipartite graph. Let $S'$ be a minimal part of $S$ which separates $T_1$ from $T \setminus T_1$ in $G$ and let $S'' = S \cap (R(T_1, S'))$. Then, $S''$ is an oct for $G[R(T_1, S')]$. 

166
Furthermore, if $|T_1| = 2$, then any oct for $G[R(T_1, S')]$ intersects all even $T_1$-paths in $G[R(T_1, S')]$.

**Lemma 7.5.9.** Let $I = (G, T = T_e \cup T_o, k)$ be an instance of BPMWC, let $S$ be an oct solution for this instance, and $T_1$ be a subset of $T_e$ such that $|T_1| \leq 2$, and $S$ is a $T_1$-$T_2$ separator where $T_2 = T \setminus T_1$. Furthermore, suppose that if $|T_1| = 2$, then the vertices in $T_1$ are adjacent. Let $\hat{S}$ be a minimal part of $S$ separating $T_1$ and $T_2$. Let $\hat{S}_1$ be a $T_1$-$T_2$ separator which well dominates $\hat{S}$. Then, there is also an oct solution for the instance which contains $\hat{S}_1$.

**Proof.** Let $\hat{K} \subseteq S$ be a minimum oct for the graph $G[R(T_1, \hat{S})]$ and let $\hat{K}_1$ be a minimum oct for the graph $G[R(T_1, \hat{S}_1)]$. Consider the set $S' = (S \setminus (\hat{S} \cup \hat{K})) \cup (\hat{S}_1 \cup \hat{K}_1)$. We claim that $S'$ is also a solution for the given instance. It is clear that $|S'| \leq |S|$. Hence, it remains to show that $S'$ is indeed a pmwc for the given instance and an oct of $G$.

Suppose that $S'$ is not a pmwc for the given instance. Then, there are two terminals $t_i$ and $t_j$ such that there is a path $P$ of forbidden parity between $t_i$ and $t_j$ in the graph $G \setminus S'$. Since $S$ is a solution, this path intersects $S \setminus S' = \hat{S} \setminus \hat{S}_1$. That is, there is a vertex $v \in \hat{S} \setminus \hat{S}_1$ such that the path $P$ intersects $v$. Suppose $t_i \in T_2$ or $t_j \in T_2$. Then, the presence of $P$ in $G \setminus S'$ implies the presence of a path from $v$ to $T_2$ in $G \setminus S'$. Since $\hat{S}_1 \subseteq S'$, there is a path from $v$ to $T_2$ in $G \setminus \hat{S}_1$. However, since $\hat{S}_1$ dominates $\hat{S}$, there is no path from $\hat{S} \setminus \hat{S}_1$ to $T_2$ in the graph $G \setminus \hat{S}_1$ (by Lemma 3.2.9), which contradicts that there is a path from $v \in \hat{S} \setminus \hat{S}_1$ to $T_2$ in the graph $G \setminus \hat{S}_1$. Hence, we conclude that $t_i, t_j \in T_1$.

Since $|T_1| \leq 2$, $T_1 = \{t_i, t_j\}$. Since $T_1 \subseteq T_e$, the path $P$ is an even $t_i$-$t_j$ path. However, $(t_i, t_j)$ is an edge (by our assumption). Therefore, by Observation 7.5.8, $\hat{K}_1$ intersects every even $T_1$ path in $G[R(T_1, \hat{S}_1)]$ and since $\hat{K}_1 \subseteq S'$, $S'$ intersects every
even $T_1$-path, a contradiction. Hence, we conclude that $S'$ is a pmwc for the given instance. The same argument also shows that $S'$ is an oct for $G$. This completes the proof of the lemma.

The above lemma describes the dominating set for the set of minimal $T_1$-$T_2$ separators and the following lemma shows that we can compute a bounded set of vertices intersecting a solution in FPT time.

**Lemma 7.5.10.** Let $(G, T = T_e \cup T_o, k)$ be an instance of BPMWC with a solution $S$. Let $T_1$ be a subset of $T_e$ such that $|T_1| \leq 2$, and $S$ is a $T_1$-$T_2$ separator where $T_2 = T \setminus T_1$. Furthermore, if $|T_1| = 2$, then the vertices in $T_1$ are adjacent. Then, there is an algorithm which runs in time $O^*(2^{O(k^2)})$ and returns a set of $2^{O(k^2)}$ vertices which intersects some oct solution for the given instance.

**Proof.** Let $X$ be a minimal part of $S$ separating $T_1$ from $T_2$. For a given set of candidate vertices, the algorithm computes (if possible) a subset of the candidate set which intersects a $T_1$-$T_2$ separator well dominating $X$. Initially, and also when the candidate set is not explicitly given, we allow this set to be the entire vertex set of the graph, and as we prune our search, we will redefine the candidate set accordingly. We first fix a hypothetical minimum oct for the graph $G[R(T_1, X)]$, say $K \subseteq S$ and guess the size of this set, say $\ell$. Formally, given a tuple $(G, Z, T_1, T_2, k, \ell)$, the algorithm returns a set of vertices $R$, such that for any $T_1$-$T_2$ separator $X$ of size at most $k$ with an oct of size $\ell$ in the graph $G[R(T_1, X)]$, there is a $T_1$-$T_2$ separator which well dominates $X$ and has a non-empty intersection with $R$. The algorithm is invoked on the input $(G, V \setminus (T_1 \cup T_2), T_1, T_2, k', \ell')$ for every $1 \leq k' \leq k$ and $1 \leq \ell' \leq k$.

**Description of algorithm.** We first check if there is a $T_1$-$T_2$ separator of size at most $k$ contained in the given subset $Z$ (see Algorithm 7.5.2). If not, then we return NO.
If there is no path from $T_1$ to $T_2$ in $G$, then we return $\emptyset$. Otherwise, we compute the tight $T_1$-$T_2$ separator sequence $I$, comprising only the vertices in $Z$. We call a $T_1$-$T_2$ separator $S'$ good if the size of the minimum oct in the graph $G[R(T_1, S')]$ is at most $l$ and we call it bad otherwise. The following observation plays a crucial role in allowing us to ignore (potentially) large parts of the graph during our search.

**Observation 7.5.11.** If a $T_1$-$T_2$ separator $S'$ is good, then all $T_1$-$T_2$ separators covered by $S'$ are also good and if $S'$ is bad, then all $T_1$-$T_2$ separators which cover $S'$ are bad.

For each $T_1$-$T_2$ separator $Y \in I$, we now determine whether $Y$ is good or bad. For this we only need check if there is an oct of size at most $\ell$ in the graph $G[R(T_1, Y)]$ and hence this requires time $O^*(2.3146^\ell)$ (by Corollary 6.5.5).

Let $P_1$ be the maximal element of $I$ which is good and let $P_2$ be the minimal element of $I$, which is bad. That is, $P_1$ is good and every separator in $I \setminus \{P_1\}$ which covers $P_1$ is bad, $P_2$ is bad and every separator in $I \setminus \{P_2\}$ covered by $P_2$ is good. If all the separators in $I$ are good, then $P_2$ is defined as $T_2$ and if all separators in $I$ are bad, then $P_1$ is defined as $T_1$. We now move on to the description of the rest of the algorithm.

- We add the vertices in $P_1 \setminus T_1$ and $P_2 \setminus T_2$ into the set $R$.

- We set $Z' = Z \cap (R(T_1, P_2) \cap NR(T_1, P_1))$ and add the vertices returned by the invocation $\text{CWDS}(G, Z', T_1, T_2, k, \ell)$ to $R$. 

169
Input: \((G, Z, T_1, T_2, k, \ell)\)

Output: A subset of \(Z\) which, for every \(T_1-T_2\) separator \(X\), intersects a \(T_1-T_2\) separator well dominating \(X\) or NO if no such separator exists

1. if \(k < 0\) then return NO
2. Check if there is a \(T_1-T_2\) separator of size at most \(k\) contained in \(Z\)
3. if there is no such separator then return NO
4. if there is no \(T_1-T_2\) path in \(G\) then return \(\emptyset\)
5. \(I = \text{TIGHT-SEPARATOR-SEQUENCE}(G, T_1, T_2, k, \ell)\)
6. for each \(Y \in I\) do
   7. \(O_Y \leftarrow \text{oct}(G[R(T_1, Y)])\)
7. \(P_1 \leftarrow \text{maximal separator with } O_Y \leq \ell\)
8. \(P_2 \leftarrow \text{minimal separator with } O_Y > \ell\)
9. \(R \leftarrow (P_1 \setminus T_1) \cup (P_2 \setminus T_2)\)
10. \(Z' = Z \cap (R(T_1, P_2) \cap NR(T_1, P_1))\)
11. \(R \leftarrow R \cup \text{CWDS}(G, Z', T_1, T_2, k, \ell)\)
12. for each ordered 4-partition \(J = (P_{nr}^1, P_e^1, \tilde{K}, P_o^1)\) of \(P_1\) do
   13. \(R \leftarrow R \cup \bigcup_{k' \leq k-1, \ell' \leq \ell} \text{CWDS}(G_J, Z \cap G_J, T_1, P_{nr}^1, k', \ell')\)
14. for each ordered 4-partition \(J = (P_{nr}^2, P_e^2, \tilde{K}, P_o^2)\) of \(P_2\) do
   15. \(R \leftarrow R \cup \bigcup_{k' \leq k-1, \ell' \leq \ell} \text{CWDS}(G_J, Z \cap G_J, T_1, P_{nr}^2, k', \ell')\)
16. return \(R\)

Algorithm 7.5.2: Algorithm COMPUTE-WELL-DOMINATING-SET (CWDS)

For each ordered 4-partition of \(P_1\), \(J = (P_{nr}^1, P_e^1, \tilde{K}, P_o^1)\) we construct a graph \(G_J\) as follows. Initially, we set \(G_J = G[R(T_1, P_1) \cup (P_1 \setminus \tilde{K})]\). For every pair in \(P_e^1 \times P_o^1\), we add an edge between the vertices and for every pair in \(P_{nr}^1 \times P_{nr}^1\), we add a subdivided edge between the vertices. This completes the construction of \(G_J\).

Now, for each \(G_J\), for each \(1 \leq k' \leq k - 1\) and for each \(0 \leq \ell' \leq \ell\), we recurse on the instance \((G_J, Z \cap G_J, T_1, P_{nr}^1, k', \ell')\) and add the vertices in the sets returned, to \(R\).

Similarly, we do the same for every ordered 4-partition of \(P_2\).
Finally, we return the set $\mathcal{R}$.

This completes the description of the algorithm. We now prove the correctness of the algorithm.

**Correctness.** For each tuple $I = (G, Z, T_1, T_2, k, \ell)$ on which the algorithm is invoked, we define a measure $\mu(I) = 2k - \lambda$ where $k$ is the upper bound on the size of the $T_1$-$T_2$ separator we are searching for in $I$, and $\lambda$ is the size of the smallest such separator. We prove the correctness of the algorithm by induction on the measure $\mu(I)$.

In the base case, if $\lambda > k$, then algorithm returns NO, which is clearly correct. Similarly, if $\lambda = 0$, then there is no path between $T_1$ and $T_2$ in the graph and hence the algorithm simply returns $\emptyset$ as the separator, which is also correct. This completes the correctness of the base cases and we now assume that the algorithm is correct on all instances $I$ with $\mu(I) < \mu$. Now, consider an instance $I$ such that $\mu(I) = \mu$.

1. Suppose $P_1$ covers the separator $X$ or $P_1 = X$. In this case, the algorithm is correct on the instance $I$ since $P_1$ itself is a $T_1$-$T_2$ separator well dominating $X$ and the set returned by $\text{CWDS}(I)$ contains $P_1$.

2. The separator $X$ covers $P_1$, and $X$ is covered by $P_2$. Let $\tilde{S}_1$ be the intersection of $X$ with $P_1$ and $\tilde{S}_2$ be the intersection of $X$ with $P_2$. Suppose that $\tilde{S}_1 \cup \tilde{S}_2$ is non-empty. But $\tilde{S}_1 \cup \tilde{S}_2$ is contained in $P_1 \cup P_2$ and is disjoint from $T_1 \cup T_2$. Therefore, since the set returned by $\text{CWDS}(I)$ contains $(P_1 \setminus T_1) \cup (P_2 \setminus T_2)$, the algorithm is correct on the instance $I$. 

171
Now, consider the case when both $\tilde{S}_1$ and $\tilde{S}_2$ are empty. Let $I'$ be the instance on which the algorithm is invoked in Step 13. By Lemma 7.5.5, any $T_1$-$T_2$ separator which lies in the set $NR(T_1, P_1) \cap R(T_1, P_2)$ has size at least $|P_1| + 1$. Hence, $\mu(I') \leq \mu - 1$ and by the induction hypothesis, $\text{CWDS}(I')$ returns a set intersecting a $T_1$-$T_2$ separator of size at most $k$ which well dominates $X$, which proves the correctness of the algorithm on the instance $I$ as well.

3. The separator $X$ is incomparable with $P_1$. Let $\tilde{S}_1$ be the intersection of $X$ with $P_1$, $P_1^r$ be the intersection of $P_1$ with $R_G(T_1, X)$, $\tilde{K} = K \cap P_1$ and $P_1^{nr} = P_1 \setminus (P_1^r \cup \tilde{K})$. Also, let $X^r$ be the intersection of $X$ with $R_G(T_1, P_1)$ and let $X^{nr}$ be the rest of $X$. Since $X$ is incomparable with $P_1$, by Observation 3.2.8, $P_1^r$, $X^r$, $P_1^{nr}$ and $X^{nr}$ are non-empty. Let $P_1^e$ and $P_1^o$ be a bipartition of $P_1^r \setminus \tilde{K}$ such that the vertices in $P_1^e$ and $P_1^o$ appear in different partitions of some bipartition of $G \setminus S$ ($G \setminus S$ is bipartite since $S$ is an oct for $G$).

Since the set returned by $\text{CWDS}(I)$ contains $P_1 \setminus T_1$, if $\tilde{S}_1$ is non-empty then the algorithm is correct on $I$. Now, consider the case when $\tilde{S}_1$ is empty. Consider the graph $G_J$ corresponding to the partition $(P_1^{nr}, P_1^e, \tilde{K}, P_1^o)$. Also, let $K^r = K \cap R(T_1, P_1)$ and let $\ell' = |K^r|$. Let $I'$ be the instance $(G_J, Z \cap G_J, T_1, P_1^{nr}, |K^r|, \ell')$. Recall that the set returned by the invocation $\text{CWDS}(I)$ contains the vertices in the set returned by the invocation $\text{CWDS}(I')$ (Line 15 of the algorithm). We now prove the following claim which will be used along with the induction hypothesis to prove the correctness of the algorithm on $I$.

Claim 7. (a) The set $X^r$ is a $T_1$-$P_1^{nr}$ separator in the graph $G_J$ such that the subgraph $G_J[R_G(J, T_1, X^r)]$ has an oct of size $\ell'$.
(b) Let $X'$ be a $T_1$-$P_1^{nr}$ separator which well dominates $X^r$ in the graph $G_J$. Then, the
set \(X' \cup X_{nr}\) well dominates \(X\) in the graph \(G\).

**Proof.** (a) In order to prove this statement, it is sufficient to show that \(K^r\) is an oct for the graph \(G' = G_J[R_{G_J}(T_1, X^r)]\). If this were not the case, then there is an odd cycle \(C\) in the graph \(G' \setminus K^r\). Replace any edge (respectively subdivided edge) of \(C\) which is present in \(G_J\), but not present in \(G\), with a corresponding path of the same parity in \(G\) (recall that this corresponding path is also present in \(G \setminus S\) to obtain a closed walk \(W\). Observe that \(W\) only contains vertices which are disjoint from \(S\). Therefore, it must be the case that \(W\) is an odd walk in the graph \(G \setminus S\). However, this contradicts our assumption that \(S\) is an oct for \(G\).

(b) Since \(X'\) well dominates \(X^r\) in \(G_J\), \(X'' = X' \cup X_{nr}\) is a \(T_1\)-\(T_2\) separator dominating \(X\) since it separates \(T_1\) from \(P^nr_1\) and \(T_2\) from \(P^r_1\) and \(|X'| \leq |X^r|\) implies that \(|X''| \leq |X|\). Therefore, it is sufficient to show that the graph \(G' = G[R(T_1, X'')]\) has an oct of size at most \(\ell\). We claim that the set \(K'' = K' \cup (K \setminus K^r)\) is indeed such an oct for \(G'\). It is clear that the size of \(K''\) is at most that of \(K\). Since \(|K| = \ell\), \(|K''| \leq \ell\). Therefore it remains to show that \(K''\) is indeed an oct for \(G'\).

If this were not the case, then there is an odd cycle \(C\) in the graph \(G' \setminus K''\). Clearly, \(C\) cannot intersect \(P^nr_1\) since \(X''\) separates \(T_1\) from \(P^nr_1\) by definition. Suppose that \(C\) lies in the set \(P^r_1 \cup NR(T_1, P_1)\). Then, \(C\) also lies in the set \(P^r_1 \cup NR(T_1, P_1)\) in the graph \(G[R(T_1, X)]\). But \(K\) is an oct for \(G[R(T_1, X)]\) and since \(K^r\) lies in \(R(T_1, P_1)\) it must be the case that \(K \setminus K^r\) intersects \(C\). However, \(K \setminus K^r\) is also contained in \(K''\), which implies that \(K''\) intersects \(C\), a contradiction. Therefore we may assume that \(C\) intersects the set \(R(T_1, P_1)\).

Now, consider any subpath of \(C\) lying in \(P^r_1 \cup NR(T_1, P_1)\) with only the endpoints in \(P^r_1\) and the remaining vertices in \(NR(T_1, P_1)\). If this path is odd, then there is an edge between these two vertices in \(G_J\) and if this path is even, then there is a subdivided
edge between these two vertices in $G_J$ (by definition of $G_J$). Therefore, we can replace all such sub-paths of $C$ with the corresponding edge or subdivided edge in $G_J$ to obtain an odd cycle $C'$ in the graph $G_J[R_{G_J}(T_1, X')]$ disjoint from the set $K'$, which is a contradiction. This completes the proof of the claim.

Now, let $X'$ be a $T_1$-$P_1^{nr}$ separator which well dominates $X^r$ in the graph $G'$. Then, due to Claim 7, $X' \cup X^{nr}$ is a $T_1$-$T_2$ separator which well dominates $X$ in the graph $G$. Therefore, a set intersecting a $T_1$-$P_1^{nr}$ separator which well dominates $X^r$ in the graph $G'$ also intersects a $T_1$-$T_2$ separator which well dominates $X$ in the graph $G$. Therefore, if the algorithm is correct on $I'$, then it is also correct on $I$ and hence it only remains to prove that the algorithm is correct on $I'$. Note that to prove that the algorithm is correct on $I'$, it is sufficient to prove that $\mu(I') < \mu(I) = \mu$ since the correctness then follows from the induction hypothesis.

By Menger’s theorem, since $P_1$ is a minimum size $T_1$-$T_2$ separator, we know that there are $P_1^{nr}$ vertex disjoint paths from $T_1$ to $P_1^{nr}$, which is also a lower bound on the size of the smallest $T_1$-$P_1^{nr}$ separator in $G_J$. Now, $\mu(I) = 2(|X^r| + |X^{nr}|) - (|P_1^{nr}| + |P_1'|)$ and $\mu(I') = 2|X^r| - |P_1^{nr}|$, which implies that $\mu(I') = \mu(I) - (2|X^{nr}| - |P_1'|)$. Since $|X^{nr}| \geq |P_1'|$, we have that $\mu(I) - \mu(I') \geq |X^{nr}|$. Since $X^{nr}$ is non empty, we conclude that $\mu(I') < \mu(I)$, which completes the proof of correctness of this case.

4. The separator $X$ is incomparable with $P_2$. This correctness of this case is analogous to the correctness of the previous case.

We note that the separator $X$ is a good separator by definition and therefore cannot cover $P_2$ due to Observation 7.5.11. Therefore the case that $X$ covers $P_2$ need not be taken into consideration and the cases we have considered are exhaustive. This completes the
proof of correctness of the algorithm.

**Running time.** We show by induction on \( \mu(I) \) that the number of vertices in the set returned by \textsc{CWDS}(I), denoted by \( N(\mu(I)) \), is bounded by \( 2^{6\mu(I)^2} \). In each of the base cases of the algorithm, we either return a single vertex or say NO, and hence the bound clearly holds. We assume that the claimed bound is true for all instances with \( \mu(I) < \mu \).

Now, consider an instance \( I \) such that \( \mu(I) = \mu \). Moreover, \( k > 1 \).

1. The number of vertices added in Step 11 is bounded by \( 2k \).

2. The number of vertices added in Step 13 is bounded by \( 2^{6(\mu-1)^2} \) by the induction hypothesis.

3. Consider the vertices added in Step 15. There are at most \( 4^k \) ordered 4-partitions of \( P_1 \), \( k \) choices for \( \ell' \), and \( k \) choices for \( k' \). For each of these choices, we return a set of size at most \( N(\mu - 1) \), which, by the induction hypothesis is at most \( 2^{6(\mu-1)^2} \). Hence, the number of vertices added in this step is bounded by \( 4^k \cdot k^2 \cdot 2^{6(\mu-1)^2} \).

4. Similarly, the number of vertices added in Step 18 is bounded by \( 4^k \cdot k^2 \cdot 2^{6(\mu-1)^2} \).

Using the fact that \( k \leq \mu \) and \( k \geq 1 \), we note that the total number of vertices returned by \textsc{CWDS}(I) is at most \( 2^{6\mu^2} \), which yields the required bound. The number of leaves of the recursion tree is clearly bounded by the size of the set returned by the algorithm, which is \( 2^{O(k^2)} \), which is also a bound on the number of internal nodes of the recursion tree. Since the algorithm spends \( O^*(2^{O(k)}) \) time at each node of the
Phase 1.

(a) If $T_e = \emptyset$ or $|T_e| = 2$ and $T_o = \emptyset$, then the problem can be solved in time $O^*(2^{O(k)})$ (Lemma 7.4.2 and Lemma 7.4.1).

(b) If neither of the above cases are true, then there is a set $T_1 \subseteq T_e$ such that $|T_1| \leq 2$ and the solution is a $T_1$-$T_2$ separator (Lemma 7.4.3).

Phase (2 + 3). There is a set $R$ of $2^{O(k^2)}$ vertices which can be computed in time $O^*(2^{O(k^2)})$ such that there is an oct solution for the given instance intersecting $R$ (Lemma 7.5.10).

Phase 4. If either case of Phase 1 applies, then solve the problem by applying Lemma 7.4.1 or Lemma 7.4.2. Otherwise, run the algorithm of Lemma 7.5.10 and branch on the vertices returned by this algorithm.

Figure 7.4: Summary of the phases of the important separator template

search tree, the total time taken by the algorithm on the given input, for a fixed $k$ and $\ell$ is $O^*(2^{O(k)})$. Since we invoke the Algorithm CWDS for every $1 \leq k' \leq k$ and $1 \leq \ell' \leq k$, the total number of vertices returned is $2^{O(k^2)}$. This completes the proof of the lemma.

The phases are summarized in Figure 7.4. It is easy to see that the running time of the algorithm for BPMWC is $O^*(2^{O(k^3)})$. This completes the proof of Theorem 7.1.1.