Theorem 6.1. *Given a* Set Cover *instance* $(U, \mathcal{F}, k)$*, the minimum possible size of a subfamily* $\mathcal{F}' \subseteq \mathcal{F}$ *that covers* U *can be found in time* $2^{|U|}(|U| + |\mathcal{F}|)^{\mathcal{O}(1)}$*.*

Proof. Let $\mathcal{F} = \{F_1, F_2, \ldots, F_{|\mathcal{F}|}\}$. We define the dynamic-programming table as follows: for every subset $X \subseteq U$ and for every integer $0 \le j \le |\mathcal{F}|$, we define $T[X, j]$ as the minimum possible size of a subset $\mathcal{F}' \subseteq \{F_1, F_2, \ldots, F_j\}$ that covers X. (Henceforth, we call such a family $\mathcal{F}'$ a *valid candidate* for the entry $T[X, j]$.) If no such subset $\mathcal{F}'$ exists (i.e., if $X \not\subseteq \bigcup_{i=1}^{j} F_i$), then $T[X, j] = +\infty$.

In our dynamic-programming algorithm, we compute all $2^{|U|}(|\mathcal{F}| + 1)$ values $T[X, j]$. To achieve this goal, we need to show (a) base cases, in our case values $T[X, j]$ for $j = 0$; (b) recursive computations, in our case how to compute the value $T[X, j]$ knowing values $T[X', j']$ for $j' < j$.

For the base case, observe that $T[\emptyset, 0] = 0$ while $T[X, 0] = +\infty$ for $X \neq \emptyset$.

For the recursive computations, let $X \subseteq U$ and $0 < j \le |\mathcal{F}|$; we show that

$$T[X, j] = \min(T[X, j-1], 1 + T[X \setminus F_j, j-1]). \tag{6.1}$$

We prove (6.1) by showing inequalities in both directions. In one direction, let $\mathcal{F}' \subseteq \{F_1, F_2, \ldots, F_j\}$ be a family of minimum size that covers X. We distinguish two cases. If $F_j \notin \mathcal{F}'$, then note that $\mathcal{F}'$ is also a valid candidate for the entry $T[X, j-1]$ (i.e., $\mathcal{F}' \subseteq \{F_1, F_2, \ldots, F_{j-1}\}$ and $\mathcal{F}'$ covers X). If $F_j \in \mathcal{F}'$, then $\mathcal{F}' \setminus \{F_j\}$ is a valid candidate for the entry $T[X \setminus F_j, j-1]$. In the other direction, observe that any valid candidate $\mathcal{F}'$ for the entry $T[X, j-1]$ is also a valid candidate for $T[X, j]$ and, moreover, for every valid candidate $\mathcal{F}'$ for $T[X \setminus F_j, j-1]$, the family $\mathcal{F}' \cup \{F_j\}$ is a valid candidate for $T[X, j]$. This finishes the proof of (6.1).

By using (6.1), we compute all values $T[X, j]$ for $X \subseteq U$ and $0 \le j \le |\mathcal{F}|$ within the promised time bound. Finally, observe that $T[U, |\mathcal{F}|]$ is the answer we are looking for: the minimum size of a family $\mathcal{F}' \subseteq \{F_1, F_2, \ldots, F_{|\mathcal{F}|}\} = \mathcal{F}$ that covers U. $\square$

We remark that, although the dynamic-programming algorithm of Theorem 6.1 is very simple, we suspect that the exponential dependency on $|U|$, that is, the term $2^{|U|}$, is optimal. However, there is no known reduction that supports this claim with the Strong Exponential Time Hypothesis (discussed in Chapter 14).

6.1.2 Steiner Tree

Let G be an undirected graph on n vertices and $K \subseteq V(G)$ be a set of *terminals.* A *Steiner tree* for K in G is a connected subgraph H of G containing K, that is, $K \subseteq V(H)$. As we will always look for a Steiner tree of minimum possible size or weight, without loss of generality, we may assume that we focus only on subgraphs H of G that are trees. The vertices of $V(H) \setminus K$ are called *Steiner vertices* of H. In the (weighted) Steiner Tree problem, we are given an undirected graph G, a weight function $\mathbf{w}\colon E(G) \to \mathbb{R}_{>0}$ and a subset of terminals $K \subseteq V(G)$, and the goal is to find a Steiner tree H for K in G whose weight $\mathbf{w}(H) = \sum_{e \in E(H)} \mathbf{w}(e)$ is minimized. Observe that if the graph G is unweighted (i.e., $\mathbf{w}(e) = 1$ for every $e \in E(G)$), then we in fact optimize the number of edges of H, and we may equivalently optimize the number of Steiner vertices of H.

For a pair of vertices $u, v \in V(G)$, by $\mathrm{dist}(u, v)$ we denote the cost of a shortest path between u and v in G (i.e., a path of minimum total weight). Let us remind the reader that, for any two vertices u, v, the value $\mathrm{dist}(u, v)$ is computable in polynomial time, say by making use of Dijkstra's algorithm.

The goal of this section is to design a dynamic-programming algorithm for Steiner Tree with running time $3^{|K|}n^{\mathcal{O}(1)}$, where $n = |V(G)|$.

We first perform some *preprocessing steps*. First, assume $|K| > 1$, as otherwise the input instance is trivial. Second, without loss of generality, we may assume that G is connected: a Steiner tree for K exists in G only if all terminals of K belong to the same connected component of G and, if this is the case, then we may focus only on this particular connected component. This assumption ensures that, whenever we talk about some minimum weight Steiner tree or a shortest path, such a tree or path exists in G (i.e., we do not minimize over an empty set). Third, we may assume that each terminal in K is of degree exactly 1 in G and its sole neighbor is not a terminal. To achieve this property, for every terminal $t \in K$, we attach a new neighbor t' of degree 1, that is, we create a new vertex t' and an edge tt' of some fixed weight, say 1. Observe that, if $|K| > 1$, then the Steiner trees in the original graph are in one-to-one correspondence with the Steiner trees in the modified graphs.

We start with defining a table for dynamic programming. For every nonempty subset $D \subseteq K$ and every vertex $v \in V(G) \setminus K$, let $T[D, v]$ be the minimum possible weight of a Steiner tree for $D \cup \{v\}$ in G.

The intuitive idea is as follows: for every subset of terminals D, and for every vertex $v \in V(G) \setminus K$, we consider the possibility that in the optimal Steiner tree H for K, there is a subtree of H that contains D and is attached to the rest of the tree H through the vertex v. For $|D| > 1$, such a subtree decomposes into two smaller subtrees rooted at some vertex u (possibly $u = v$), and a shortest path between v and u. We are able to build such subtrees for larger and larger sets D through the dynamic-programming algorithm, filling up the table $T[D, v]$.

The base case for computing the values $T[D, v]$ is where $|D| = 1$. Observe that, if $D = \{t\}$, then a Steiner tree of minimum weight for $D \cup \{v\} = \{v, t\}$

is a shortest path between v and t in the graph G. Consequently, we can fill in $T[\{t\}, v] = \mathrm{dist}(t, v)$ for every $t \in K$ and $v \in V(G) \setminus K$.

In the next lemma, we show a recursive formula for computing the values $T[D, v]$ for larger sets D.

Lemma 6.2. *For every $D \subseteq K$ of size at least 2, and every $v \in V(G) \setminus K$, the following holds*

$$T[D, v] = \min_{\substack{u \in V(G) \setminus K \\ \emptyset \neq D' \subsetneq D}} \{T[D', u] + T[D \setminus D', u] + \mathrm{dist}(v, u)\}. \tag{6.2}$$

Proof. We prove (6.2) by showing inequalities in both directions.

In one direction, fix $u \in V(G)$ and $\emptyset \neq D' \subsetneq D$. Let H_1 be the tree witnessing the value $T[D', u]$, that is, H_1 is a Steiner tree for $D' \cup \{u\}$ in G of minimum possible weight. Similarly, define H_2 for the value $T[D \setminus D', u]$. Moreover, let P be a shortest path between v and u in G. Observe that $H_1 \cup H_2 \cup P$ is a connected subgraph of G that contains $D \cup \{v\}$ and is of weight

$$\mathbf{w}(H_1 \cup H_2 \cup P) \leq \mathbf{w}(H_1) + \mathbf{w}(H_2) + \mathbf{w}(P) = T[D', u] + T[D \setminus D', u] + \mathrm{dist}(v, u).$$

Thus

$$T[D, v] \leq \min_{\substack{u \in V(G) \setminus K \\ \emptyset \neq D' \subsetneq D}} \{T[D', u] + T[D \setminus D', u] + \mathrm{dist}(v, u)\}.$$

In the opposite direction, let H be a Steiner tree for $D \cup \{v\}$ in G of minimum possible weight. Let us root the tree H in the vertex v, and let u_0 be the vertex of H that has at least two children and, among such vertices, is closest to the root. An existence of such a vertex follows from the assumptions that $|D| \geq 2$ and that every terminal vertex is of degree 1. Moreover, since every terminal of K is of degree 1 in G, we have $u_0 \notin K$. Let u_1 be an arbitrarily chosen child of u_0 in the tree H. We decompose H into the following three edge-disjoint subgraphs:

1. P is the path between u_0 and v in H;
2. H_1 is the subtree of H rooted at u_1, together with the edge $u_0 u_1$;
3. H_2 consists of the remaining edges of H, that is, the entire subtree of H rooted at u_0, except for the descendants of u_1 (that are contained in H_1). See Fig. 6.1.

Let $D' = V(H_1) \cap K$ be the terminals in the tree H_1. Since every terminal is of degree 1 in G, we have $D \setminus D' = V(H_2) \cap K$. Observe that, as H is of minimum possible weight, $D' \neq \emptyset$, as otherwise $H \setminus H_1$ is a Steiner tree for $D \cup \{v\}$ in G. Similarly, we have $D' \subsetneq D$ as otherwise $H \setminus H_2$ is a Steiner tree for $D \cup \{v\}$ in G. Furthermore, note that from the optimality of H it follows that $\mathbf{w}(H_1) = T[D', u_0]$, $\mathbf{w}(H_2) = T[D \setminus D', u_0]$ and, moreover, P is a shortest path between u_0 and v. Consequently,

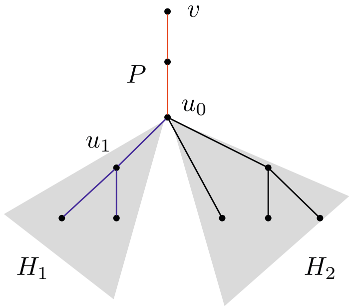

Fig. 6.1: Decomposition of H

$$\begin{aligned} T[D,v] = \mathbf{w}(H) &= T[D',u_0] + T[D\setminus D',u_0] + \mathrm{dist}(v,u_0) \\ &\geq \min_{\substack{u\in V(G)\setminus K \\ \emptyset\neq D'\subsetneq D}} \{T[D',u] + T[D\setminus D',u] + \mathrm{dist}(v,u)\}\,. \end{aligned}$$

This finishes the proof of the lemma. □

With the insight of Lemma 6.2, we can now prove the main result of this section.

Theorem 6.3. Steiner Tree *can be solved in time* $3^{|K|}n^{\mathcal{O}(1)}$.

Proof. Let (G,w,K) be an instance of Steiner Tree after the preprocessing steps have been performed. We compute all values of $T[D,v]$ in the increasing order of the cardinality of the set D. As discussed earlier, in the base case we have $T[\{t\},v] = \mathrm{dist}(t,v)$ for every $t\in K$ and $v\in V(G)\setminus K$. For larger sets D, we compute $T[D,v]$ using (6.2); note that in this formula we use values of $T[D',u]$ and $T[D\setminus D',u]$, and both D' and $D\setminus D'$ are proper subsets of D. In this manner, a fixed value $T[D,v]$ can be computed in time $2^{|D|}n^{\mathcal{O}(1)}$. Consequently, all values $T[D,v]$ are computed in time

$$\sum_{v\in V(G)\setminus K}\sum_{D\subseteq K} 2^{|D|}n^{\mathcal{O}(1)} \leq n\sum_{j=2}^{|K|}\binom{|K|}{j}2^j n^{\mathcal{O}(1)} = 3^{|K|}n^{\mathcal{O}(1)}.$$

Finally, observe that, if the preprocessing steps have been performed, then any Steiner tree for K in $V(G)$ needs to contain at least one Steiner point and, consequently, the minimum possible weight of such a Steiner tree equals $\min_{v\in V(G)\setminus K} T[K,v]$. □

We will see in Section 10.1.2 how the result of Theorem 6.3 can be improved.