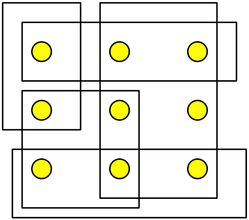
Dynamic Programming

Set Cover



- **Input :** A universe U of n elements and a family $\mathcal{F} = \{F_1, F_2, \ldots F_m\}$ of m subsets of U.
- **Parameter :** n (Size of the Universe)
- **Output :** A subfamily $\mathcal{F}' \subseteq \mathcal{F}$ of minimum size that "covers U"

$$\bigcup_{F \in \mathcal{F}'} F = U$$

Theorem

Set Cover *is FPT parameterized by the size of the universe*

Running time: $2^n \cdot \mathsf{poly}(n, m)$

Set Cover: Dynamic Programming

- Fix an ordering of the family $\mathcal{F}$: $F_1, F_2, \ldots, F_m$
- Dynamic Programming Table,

$$\text{for every } X \subseteq U \text{ and } j \in \{0, 1, \ldots, m\}$$

$$T[X, j] = \text{ size of a min subset of } \{F_1, F_2, \ldots, F_j\} \text{ that covers } X$$

- Table Size: $2^{|U|} \times m$
- **Base Case :** $T[X, 0] = 0$ if $X = \emptyset$, else it is ∞.
- **Recursize Step :**

$$T[X, j] = \min \left\{ T[X, j-1],\;\; 1 + T[X \setminus F_j, j-1] \right\}$$

  - Either X can be covered using within $\{F_1, F_2, \ldots, F_{j-1}\}$
  - Or we need F_j + best solution of $X \setminus F_j$
- $T[U, m]$ is the minimum set cover size
- Maintain a candidate solution along with each $T[X, j]$.

Set Cover: Dynamic Programming

- Fix an ordering of the family $\mathcal{F}$: $F_1, F_2, \ldots, F_m$
- Dynamic Programming Table,

$$\text{for every } X \subseteq U \text{ and } j \in \{0, 1, \ldots, m\}$$

$$T[X, j] = \text{ size of a min subset of } \{F_1, F_2, \ldots, F_j\} \text{ that covers } X$$

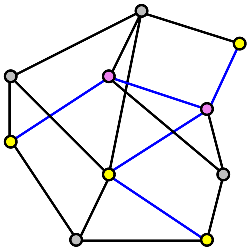
- Table Size: $2^{|U|} \times m$
- **Base Case :** $T[X, 0] = 0$ if $X = \emptyset$, else it is ∞.
- **Recursize Step :**

$$T[X, j] = \min \left\{ T[X, j-1], \;\; 1 + T[X \setminus F_j, j-1] \right\}$$

  - Either X can be covered using within $\{F_1, F_2, \ldots, F_{j-1}\}$
  - Or we need F_j + best solution of $X \setminus F_j$
- $T[U, m]$ is the minimum set cover size
- Maintain a candidate solution along with each $T[X, j]$.

Exercise: Prove that $T[X, j]$ indeed contains a minimum set cover of X from $\{F_1, F_2, \ldots, F_j\}$

Steiner Tree



- **Input :** Graph G on n vertices, $S \subseteq V(G)$ of k vertices called Terminals.
- **Parameter :** k (Number of Terminals)
- **Output :** Minimum connected subgraph H of G that contains all of S.

Observation : H must be a Tree

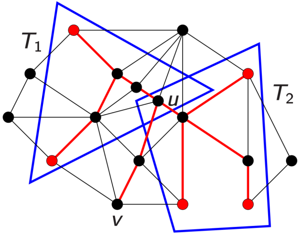
Theorem

Steiner Tree *can be solved in time* $3^k \cdot \mathsf{poly}(n)$.

Notation:

- $d_G(u, v) =$ length of shortest path between u and v in G.
- Assume every terminal $s \in S$ has degree 1

Steiner Tree: Dynamic Programming



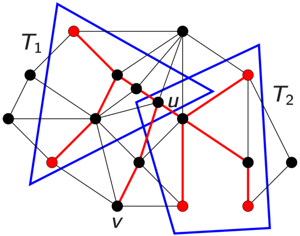
- DP Table: For $X \subseteq S$ and $v \in V(G)$

  $T[X, v] =$ minimum cost of a sub-tree containing $X \cup v$.

- Table Size: $2^S \cdot n$
- Base Case I: $T[\emptyset, v] = 1$ for every $v \in V(G)$
- Base Case II: $T[\{s\}, v] = d_G(s, v)$ for every $s \in S$
- Recursive Case: for $X \subseteq V(G)$, $|X| \geq 2$

$$T[X, v] = \min_{u \in V(G),\ \ \emptyset \neq Y \subsetneq X} d_G(u, v) + T[Y, u] + T[X \setminus Y, u]$$

Steiner Tree: Dynamic Programming



- DP Table: For $X \subseteq S$ and $v \in V(G)$

  $T[X, v] =$ minimum cost of a sub-tree containing $X \cup v$.

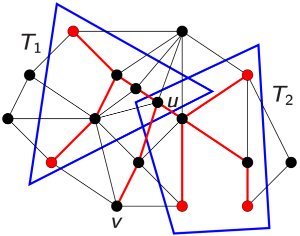
- Recursive Case:

$$T[X, v] = \min_{u \in V(G),\ \emptyset \neq Y \subsetneq X} d_G(u, v) + T[Y, u] + T[X \setminus Y, u]$$

Correctness: (LHS $\leq$ RHS)

- For any $Y \subseteq X$ and $u \in V(G)$, the RHS is the cost of a sub-tree connecting $X \cup v$.
- RHS = min-cost subtree for $Y \cup u$ + min-cost subtree for $(X \setminus Y) \cup u$ + shortest path between u and v

Steiner Tree: Dynamic Programming



- DP Table: For $X \subseteq S$ and $v \in V(G)$

  $T[X, v] =$ minimum cost of a sub-tree containing $X \cup v$.

- Recursive Case:

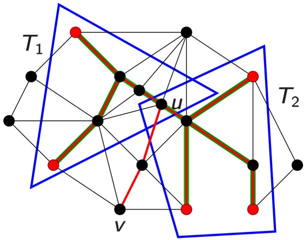
$$T[X, v] = \min_{u \in V(G),\ \emptyset \neq Y \subsetneq X} d_G(u, v) + T[Y, u] + T[X \setminus Y, u]$$

Correctness: (LHS $\geq$ RHS)

- Consider a minimum subtree H of G connecting $X \cup v$.
- root H at v, and u is the closest descendant with multiple children $\{u_1, u_2, \ldots, u_\ell\}$

Note: u exists because $|X| \geq 2$ and all terminals have degree 1.
Further $d_H(u, v) = d_G(u, v)$, by choice of H

Steiner Tree: Dynamic Programming



- DP Table: For $X \subseteq S$ and $v \in V(G)$

  $T[X, v] =$ minimum cost of a sub-tree containing $X \cup v$.

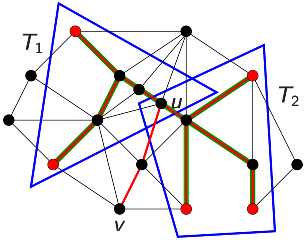
- Recursive Case:

$$T[X, v] = \min_{u \in V(G),\ \emptyset \neq Y \subsetneq X} d_G(u, v) + T[Y, u] + T[X \setminus Y, u]$$

Correctness: (LHS $\geq$ RHS)

- Let Y = all terminal from X in sub-tree of u_1.
- Split H into 3 parts
  - The sub-path between u and v
  - The sub-tree of H rooted at u_1 + edge (u, u_1)
  - The sub-tree of H excluding the above

Steiner Tree: Dynamic Programming



- DP Table: For $X \subseteq S$ and $v \in V(G)$

  $T[X, v] =$ minimum cost of a sub-tree containing $X \cup v$.

- Recursive Case:

$$T[X, v] = \min_{u \in V(G),\ \ \emptyset \neq Y \subsetneq X} d_G(u, v) + T[Y, u] + T[X \setminus Y, u]$$

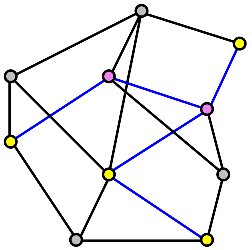
Running Time:

- Computing $T[X, v]$ requires $2^{|X|} \cdot \mathsf{poly}(n)$ time.
- Computing the entire table requires time:

$$\sum_{v \in V(G), X \subseteq S} 2^{|X|} \cdot \mathsf{poly}(n)$$

- This is $3^{|S|} \cdot \mathsf{poly}(n)$

Steiner Tree



- **Input :** Graph G on n vertices, $S \subseteq V(G)$ of k vertices called Terminals.
- **Parameter :** k (Number of Terminals)
- **Output :** Minimum connected subgraph H of G that contains all of S.

Observation : H must be a Tree

Theorem

Steiner Tree *can be solved in time* $3^k \cdot \mathsf{poly}(n)$.

Exercise: Steiner Tree with weights (Positive Integers)