20.1 Steiner Forest Problem

- **Input**
  - a graph \( G = (V, E) \)
  - cost function \( c : E \to \mathbb{R} \)
  - pairs \((s_1, t_1), \ldots, (s_k, t_k)\) of nodes

- **Feasible solution**
  \( \mathcal{F} \subseteq E \) such that \((V, \mathcal{F})\) contains an \(s_i\)-\(t_i\) path \(\forall i \in [k]\).

- **Objective**
  \[ \min \sum_{e \in \mathcal{F}} c(e) \]

20.2 Linear Program

**Definition 20.2.1** Let \( S_i = \{ S \subseteq V : |S \cap \{s_i, t_i\}| = 1\} \). And let \( \mathcal{S} = \bigcup_{i=1}^{k} S_i \).

\[
\begin{align*}
\text{minimize:} & \quad \sum_{e \in E} c(e) x_e \\
\text{subject to:} & \quad \sum_{e \in \delta(S)} x_e \geq 1 \quad \forall S \in \mathcal{S} \\
& \quad x_e \geq 0 \quad \forall e \in E
\end{align*}
\]

20.3 Dual

\[
\begin{align*}
\text{maximize:} & \quad \sum_{S \in \mathcal{S}} y_S \\
\text{subject to:} & \quad \sum_{S \in \mathcal{S}, e \in \delta(S)} y_S \leq c(e) \quad \forall e \in E \\
& \quad y_S \geq 0 \quad \forall S \in \mathcal{S}
\end{align*}
\]

20.4 Algorithm
Algorithm 1

\[ F_1 = \emptyset, \vec{y} = \vec{0}, j = 1. \]

while \( F_j \) not feasible do
    Let \( C_j = \{ S \in S : S \text{ component of } (V, F_j) \} \) be the components of \( (V, F_j) \) that are also sets in \( S \).
    Increase all \( y_S : S \in C_j \) uniformly until \( \exists e_j \in \delta(S), S \in C_j \) such that constraint for \( e_j \) is tight, i.e. \( \sum_{S \in S : e_j \in \delta(S)} y_S = c(e_j) \).
    Let \( \delta_j \) be amount of dual variables increased.
    \( F_{j+1} = F_j \cup \{ e_j \} \).
    \( j = j + 1. \)
end while

\( F = F_j. \)

while \( \exists e \in F \) such that \( F - \{ e \} \) is feasible do
    Remove \( e \) from \( F \).
end while

return \( F. \)

20.5 Properties

Note: \( \vec{y} \) is always dual feasible.

Proof: \( \vec{y} = \vec{0} \) is feasible at the beginning. At each iteration, we will increase \( y_S \) until some constraint is tight for \( e \in E \). And such \( e \) will be inside the component in the following iterations so its dual constraint will remain tight (not violated).

Note: This algorithm is polytime.

Proof: There are at most \( |E| \) iterations and at most \( n \) active components. So there are at most \( n|E| \) nonzero dual variables.

Note: Final pruning is necessary.

Proof: Consider the star graph where \( s_1 \) is in the center connected to \( v_1, \cdots, v_{n-2} \) with costs all \( 1 \) and connected to \( t_1 \) with cost \( 3 \). Then without the final pruning, the algorithm would buy the entire star rather than just the \( \{ s_1, t_1 \} \) edge.

Lemma 20.5.1 Let \( T \) be a tree. If \( S \subseteq V(T) \) such that all leaves are in \( S \), then

\[ \sum_{v \in S} \text{deg}_T(v) \leq 2|S|. \]
Proof:

\[
\sum_{v \in S} \deg_T(v) = \sum_{v \in T} \deg_T(v) - \sum_{v \notin S} \deg_T(v) = 2(n - 1) - \sum_{v \notin S} \deg_T(v) \leq 2(n - 1) - 2(n - |S|) = 2|S| - 2.
\]

Lemma 20.5.2 At all iterations \(j\), \(\sum_{S \in C_j} |F \cap \delta(S)| \leq 2|C_j|\).

Proof: Note that by construction, \(F_j\) is a forest for all \(j\) (we only ever add edges that leave a component).

Consider time \(j\) the new graph \(G_j = (V_j, E_j)\). Here \(V_j\) is the components of \((V, F_j)\), and \(E_j = \{S_1, S_2\}, S_1, S_2 \in V_j \) and \(\exists e \in F_j\) with 1 endpoint in \(S_1\), other in \(S_2\).

Equivalently, start with \((V, F_j^*)\), where \(j^*\) is the final iteration before pruning,
- Contract edges in \(F_j\);
- Remove edges in \(F_j^* - F\).

Claim 20.5.3 If \(S \in V_j\) has degree 1 in \(G_j\), then \(S \in C_j\).

Proof: Suppose \(S \notin C_j\). Let \(e \in F\) be edge incident on \(S\) in \(G\) (such an edge must exist since \(S\) has degree 1 in \(G_j\)). Then \(S \notin S\) indicates that \(S\) does not separate any \(s_i\) and \(t_i\). So pruning would have removed \(e\) from \(F\).

By Lemma 20.5.1 we finish the proof.

Claim 20.5.4 \(\sum_{S \in S} |\delta(S) \cap F|y_S \leq 2\sum_{S \in S} y_S\).

Proof: Incudtion on iterations.
At \(j = 0\), \(y_S = 0\) so it is true.
At iteration \(j\), LHS increases by \(\sum_{S \in C_j} \Delta_j |\delta(S) \cap F| \leq 2\Delta_j |C_j|\) by Lemma 20.5.2
RHS increases by \(2\sum_{S \in C_j} \Delta_j = 2\Delta_j |C_j|\).
So induction holds.

Theorem 20.5.5 Primal-dual algorithm is a 2-approximation.
Proof: By the Claim 20.5.4 above,

\[ \sum_{e \in F} c(e) = \sum_{e \in F} \sum_{S \in \delta(e)} y_S \]

\[ = \sum_{S \in \delta} |\delta(S) \cap F| y_S \]

\[ \leq 2 \sum_{S \in \delta} y_S \]

\[ \leq 2 \text{OPT}. \]

The last inequality is by weak duality.

20.6 Open Question

Is there a less than 2-approximation algorithm for Steiner Forest?

**Definition 20.6.1** \( l \)-Steiner Forest is the case to choose \( l \) of the \( k \) pairs to connect.

**Theorem 20.6.2** \( l \)-Steiner Forest has \( O(\sqrt{n}) \)-approximation.

**Theorem 20.6.3** If \( c(e) = 1 \) for all \( e \in E \), then \( l \)-Steiner Forest has \( O(n^{\frac{1}{3}(7-4\sqrt{2})}) \)-approximation.