Graph Definitions

Definition 1. *An* **undirected** *graph* G *is a pair* (V, E) *where*

- V *is the set of vertices,*
- $E \subseteq V^2$ *is the set of edges (unordered pairs)*

$$E \;=\; \{(u, v) \mid u, v \in V\}.$$

In a **directed** *graph the edges have directions (ordered pairs).*

A **weighted** *graph includes a weight function*

$$w : E \rightarrow R$$

attaching a value (weight) to each edge.

Definition 2. *A path in a graph* $G = (V, E)$ *is a sequence of vertices* $v_1, v_2, ..., v_k$ *such that for* $1 \leq i \leq k - 1$*,* $(v_i, v_{i+1}) \in E$*.*

Definition 3. *A cycle in a graph* $G = (V, E)$ *is a path* $v_1, v_2, ..., v_k$ *such that* $(v_k, v_1) \in E$*.*

Definition 4. *A tree is a graph with no cycles.*

Definition 5. *A graph* $H = (V', E')$ *is a* **subgraph** *of* $G = (V, E)$ *iff* $V' \subseteq V$ *and* $E' \subseteq E$*. A* **spanning** *subgraph if* $V' = V$*.*

Graph Representation

Adjacency List: A link list for each vertex. The list contains a pointer to each neighbor of the vertex. (and a weight for weighted graph.)

Total space $O(V + E)$.

Sequential (linear) access.

Adjacency Matrix: A $|V| \times |V|$ array, $A[i, j] = 1$ iff (i, j) is an edge in the graph, otherwise $A[i, j] = 0$. ($A[i, j] = w_{i,j}$ for weighted graph.)

Total space $O(V^2)$.

Random access.

Spanning Tree

Given a graph $G = (V, E)$ a **spanning tree** T in G is a subgraph of G that

- is connected;
- includes all the vertices of G;
- has no cycles;

Spanning Tree Characterization

Theorem 1. *A* **spanning tree** T *in a connected graph* $G = (V, E)$ *is a maximal subgraph of* G *that is a tree (i.e. any edge added to* T *closes a cycle).*

Proof. Proof by contradiction:

Assume that T is a spanning tree in G, and adding edge (v, u) to T does not close a cycle.

There are paths from w to v, and from w to u using only edges of T (since it's a spanning tree).

Thus, the path w to v, the edge (v, u) and the path u to w must include a cycle. $\square$

The Size of a Spanning Tree

Lemma 1. *A spanning tree of a connected graph* $G = (V, E)$ *has* $|V| - 1$ *edges.*

Proof.

Let T be a spanning tree of G. Fix a vertex w in T. There is a **unique** path from w to any other vertex in T.

For all $u \in V$ If an edge (u, v) was the last edge on the path from w to u direct the edge to u.

This process gives a unique direction to each edge.

Associate an edge with the vertex it is pointing to.

We have a $1 - 1$ correspondence between the edges of T and the set $V - \{w\}$. $\square$

Spanning Tree Algorithm

Spanning_Tree(G)

1. $VT \leftarrow \emptyset$;

2. $ET \leftarrow \emptyset$;

3. For $i = 1$ to $|E|$ do

   3.1 If $\{e_i\} \cup ET$ has no cycles then
   3.1.1. $VT \leftarrow VT \cup e_i$;
   3.1.2. $ET \leftarrow ET \cup \{e_i\}$;

Correctness

Theorem 2. *The algorithm computes a spanning tree in G.*

Proof. The algorithm generates a maximal tree subgraph of G. $\square$

Minimum Weighted Spanning Tree

Given a graph $G = (V, E)$ and a **weight** function $w : E \to R$ (on the edges), a weight of a spanning tree $T = (V, E(T))$ is

$$w(T) \;=\; \sum_{e \in E(T)} w(e).$$

A minimum spanning tree of G with weight function w, is a spanning tree of G with minimum weight.

Theorem 3. *Let $T = (V(T), E(T))$ be a minimum spanning tree of G. Let $S = (V(S), E(S))$ be a subgraph of T. Let e be an edge of minimum weight among all edges that do not close a cycle with edges in $E(S)$, then $E(S) \cup \{e\})$ is a subgraph of a minimum spanning tree of G.*

Proof. If $e \in E(T)$ we are done.

Assume that $e \notin E(T)$. Add the edge e to T, we get a cycle with at least one edge $e' \notin E(S)$, thus $w(e') \geq w(e)$.

Let $T' = (V, E(T) + \{e\} - \{e'\})$.

T' is a spanning tree, and $w(T') \leq w(T)$, therefore it is also a minimum spanning tree.

$\square$

Minimum Spanning Tree Algorithm

Minimum Spanning Tree (G, w)

1. $Q \leftarrow E$;
2. $E(T) \leftarrow \emptyset$;
3. $V(T) \leftarrow \emptyset$;
4. For $i = 1$ to $V - 1$ do

4.1 $e \leftarrow Min(Q)$;
4.2 $E(T) \leftarrow E(T) \cup \{e\}$;
4.3 $V(T) \leftarrow V(T) \cup$ vertices of e.;
4.4 $Q \leftarrow Q - \{e\}$;
4.5 For all $e \in Q$
4.5.1. If $e \cup E(T)$ close a cycle then $Q \leftarrow Q - \{e\}$;

Greedy Algorithm

A **greedy** algorithm makes a sequence of local optimal choices.

Under various conditions greedy construction is optimal, i.e. local optimal choices lead to global optimal solution, and there is no need for backtracking - but not always!

Analysis

Theorem 4. *The algorithm constructs a minimum spanning tree.*

Proof. We prove by induction on the size of the set $E(T)$, that $E(T)$ is always a subset of a minimum spanning tree.

The induction hypothesis holds for $E(T) = \emptyset$.

Assume that it holds for $|E(T)| = i - 1$, then by the above theorem and the choice of the algorithm the claim holds for $|E(T)| = i$.

Thus, also holds for $|E(T)| = |V| - 1$. $\square$