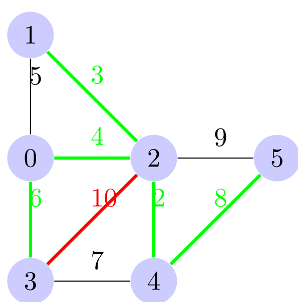
1 O notation

$\geq \Omega$
$= \Theta$
$\leq O$

2 Minimum Spanning Tree

Input: Connected undirected weighted $G = (V, E) \quad W : E \to \mathbb{R}$
Output: Subset of edges $T \subseteq E$ that connects all vertices and $min\ w(e)\,, e \in T$.
(T = tree)

2.1 Example



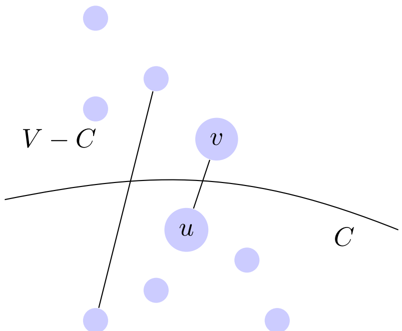
The edge between 2 and 3 with a weight of 10 is an edge to definitely remove because it is unnecessary (since there is a cycle) and it also weighs a lot.

Green good edges to keep as their weights are small.
Red not so good due to large weights.

2.2 Applications:

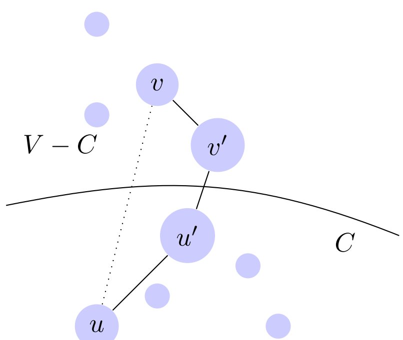
- network design
- clustering
- vision
- approximation algorithms

2.3 Cut Lemma



Lemma 2.1. *Let $C \subseteq V$ $(u, v) \in E$, $v \notin C$, $u \in C$. (u, v) is a lightest edge of this kind. There there exists a minimum spanning tree that contains (u, v).*

Proof. Suppose T is a MST that does not contain (u, v).



T connected u and v

$\Rightarrow$ There must exist an edge $(u', v') \in T$ $u' \in C$, $v' \notin C$ $\quad w(u', v') \geq w(u, v)$ Thus, weight of $T \bigcup \{(u, v)\} - \{(u', v')\} \triangleq T'$ is no larger than T's weight.

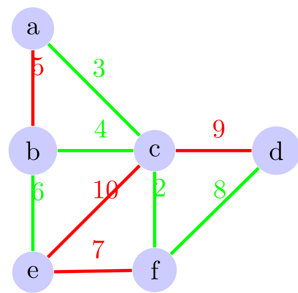
Note 2.1. Note that T' is a tree that connects all vertices. Because a path $a \rightsquigarrow u' \to v' \rightsquigarrow b$ in T becomes a path $a \rightsquigarrow u' \rightsquigarrow u \to v \rightsquigarrow v' \rightsquigarrow b$

$\square$

Note 2.2. Observation: If all edge weights are distinct, the MST is unique, because the weight of T' is strictly smaller than the weight of T

3 Prim's Algorithm

Chose the shortest edge when in a cut.



Notice that the edges we choose here are the same as in the previous example.

3.1 Algorithm

Please refer Algorithm 1. $\Pi(u)$ holds the parent for u. The Tree T is constructed using edge $(\Pi(u), u)$.

Algorithm 1 Prim's Algorithm

procedure PRIM'S ALGORITHM(G, s) ▷ $G(V, E)$ weighted graph w/non-negative weights, $s \in V$, s is an arbitary vertex in V
    $Q \leftarrow V$ ▷ Q is a priority queue containing all vertices.
    $\forall v \in V - \{s\}$, $\quad key(v) \leftarrow \infty$ ▷ for all vertices except for $s($
    $key(s) \leftarrow 0$ ▷ s is an arbitrary vertex
    $T \leftarrow \{\}$
    while $Q \neq 0$ **do**
        $u \leftarrow$Extract-Min(Q) ▷ The edge we take to T is $(\Pi(u), u)$
        $T \leftarrow T \bigcup (\Pi(u), u)$
        for all neighbor v of u **do**
            if $v \in Q$ **then**
                Decrease-Key$(Q, v, w(u, v))$ ▷ if $key(v) \geq w(u, v)$ then $key(v) = w(u, v)$
                $\Pi(v) \leftarrow u$
            end if
        end for
    end while
    return MST for G
end procedure

3.2 Run Time

$m = |E|$ (# of edges)
$n = |V|$ (# of nodes)
Runtime: $O(n \log n + m \log n)$
Runtime does not change on where you start.