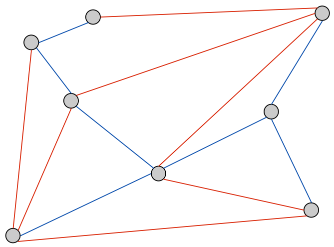
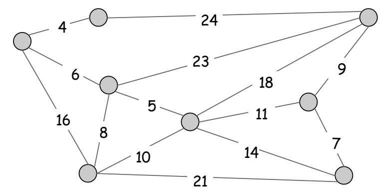
MST: Red Rule, Blue Rule

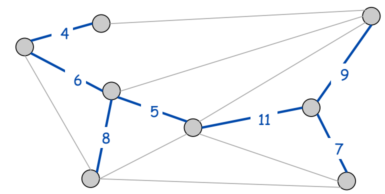


Minimum Spanning Tree

Minimum spanning tree. Given a connected graph G with real-valued edge weights c_e, an *MST* is a spanning tree of G whose sum of edge weights is minimized.



$G = (V, E)$



$T = (V, F)$
$w(T) = 50$

Cayley's Theorem (1889). There are n^{n-2} spanning trees of K_n.

↑ can't solve by brute force

Minimum Spanning Tree Origin

Otakar Boruvka (1926).

- Electrical Power Company of Western Moravia in Brno.
- Most economical construction of electrical power network.
- Concrete engineering problem is now a cornerstone problem in combinatorial optimization.

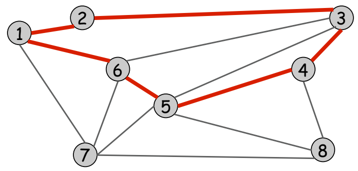
Applications

MST is fundamental problem with diverse applications.

- Network design.
  - telephone, electrical, hydraulic, TV cable, computer, road
- Approximation algorithms for NP-hard problems.
  - traveling salesperson problem, Steiner tree
- Indirect applications.
  - max bottleneck paths
  - LDPC codes for error correction
  - image registration with Renyi entropy
  - learning salient features for real-time face verification
  - reducing data storage in sequencing amino acids in a protein
  - model locality of particle interactions in turbulent fluid flows
  - autoconfig protocol for Ethernet bridging to avoid cycles in a network
- Cluster analysis.

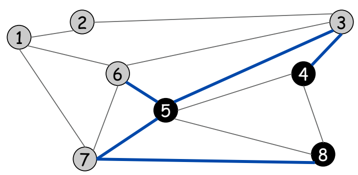
Cycles and Cuts

Cycle. Set of edges the form a-b, b-c, c-d, …, y-z, z-a.



Cycle = 1-2, 2-3, 3-4, 4-5, 5-6, 6-1

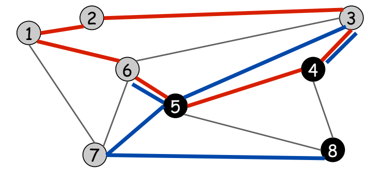
Cut. The cut induced by a subset of nodes S is the set of all edges with exactly one endpoint in S.



S = { 4, 5, 8 }
Cut = 5-6, 5-7, 3-4, 3-5, 7-8

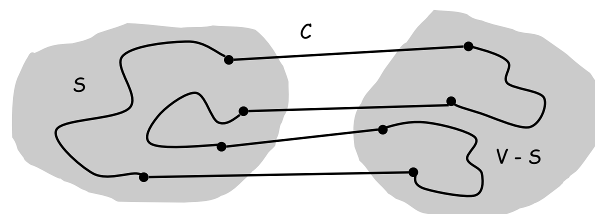
Cycle-Cut Intersection

Claim. A cycle and a cut intersect in an even number of edges.



Cycle = 1-2, 2-3, 3-4, 4-5, 5-6, 6-1
Cut = 3-4, 3-5, 5-6, 5-7, 7-8
Intersection = 3-4, 5-6

Pf. (by picture)



Generic MST Algorithm

Red rule. Let C be a cycle with no red edges. Select an uncolored edge of C of max weight and color it red.

Blue rule. Let D be a cut with no blue edges. Select an uncolored edge in D of min weight and color it blue.

Greedy algorithm. Apply the red and blue rules (non-deterministically!) until all edges are colored.

↑ can stop once n-1 edges colored blue

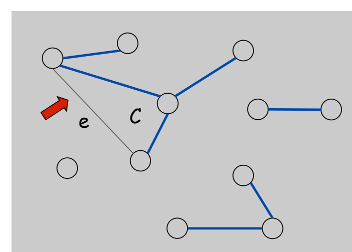
Theorem. The blue edges form a MST.

Reference: *Data Structures and Algorithms* by R. E. Tarjan

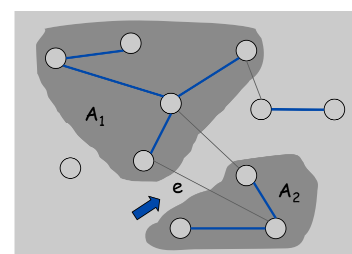
Greedy Algorithm: Proof of Correctness

Claim. The greedy algorithm terminates.
Pf. (by contradiction)

- Suppose edge e is left colored; let's see what happens.
- Blue edges form a forest F.
- Case 1: adding e to F creates a cycle C.
- Case 2: adding e to F connects two components A_1 and A_2. ▪



Case 1: apply red rule to cycle C and color e red.



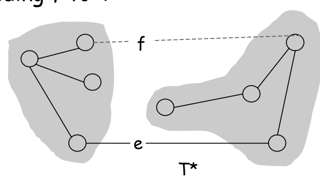
Case 2: apply blue rule to A_1 or A_2, and color some edge blue.

Greedy Algorithm: Proof of Correctness

Theorem. Upon termination, the blue edges form a MST.
Pf. (by induction on number of iterations)

> Color Invariant: There exists a MST T* containing all the blue edges and none of the red ones.

- Base case: no edges colored $\Rightarrow$ every MST satisfies invariant.
- Induction step: suppose color invariant true before blue rule.
  - let D be chosen cut, and let f be edge colored blue
  - if $f \in T^*$, T* still satisfies invariant
  - o/w, consider fundamental cycle C by adding f to T*
  - let e be another edge in $C \cap D$
  - e is uncolored and $c_e \geq c_f$ since
    - $e \in T^* \Rightarrow$ e not red
    - blue rule $\Rightarrow$ e not blue, $c_e \geq c_f$
  - $T^* \cup \{ f \} - \{ e \}$ satisfies invariant

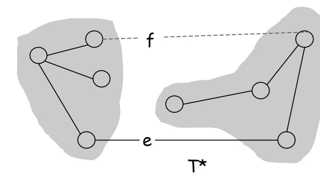


Greedy Algorithm: Proof of Correctness

Theorem. Upon termination, the blue edges form a MST.
Pf. (by induction on number of iterations)

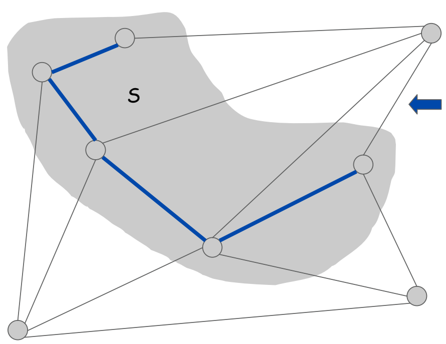
> Color Invariant: There exists a MST T* containing all the blue edges and none of the red ones.

- Induction step (cont): suppose color invariant true before red rule.
  - let C be chosen cycle, and let e be edge colored red
  - if $e \notin T^*$, T* still satisfies invariant
  - o/w, consider fundamental cut D by deleting e from T*
  - let f be another edge in $C \cap D$
  - f is uncolored and $c_e \geq c_f$ since
    - $f \notin T^* \Rightarrow$ f not blue
    - red rule $\Rightarrow$ f not red, $c_e \geq c_f$
  - $T^* \cup \{ f \} - \{ e \}$ satisfies invariant ▪



Special Case: Prim's Algorithm

Prim's algorithm. [Jarník 1930, Dijkstra 1957, Prim 1959]
- S = vertices in tree connected by blue edges.
- Initialize S = any node.
- Apply blue rule to cut induced by S.



Implementation: Prim's Algorithm

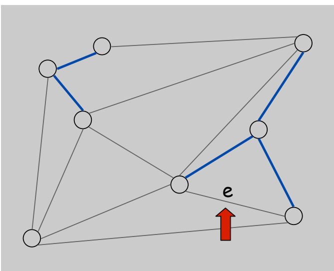
Implementation. Use a priority queue ala Dijkstra.
- Maintain set of explored nodes S.
- For each unexplored node v, maintain attachment cost a[v] = cost of cheapest edge v to a node in S.
- $O(n^2)$ with an array; $O(m \log n)$ with a binary heap.

```
Prim(G, c) {
   foreach (v ∈ V) a[v] ← ∞
   Initialize an empty priority queue Q
   foreach (v ∈ V) insert v onto Q
   Initialize set of explored nodes S ← φ

   while (Q is not empty) {
      u ← delete min element from Q
      S ← S ∪ { u }
      foreach (edge e = (u, v) incident to u)
          if ((v ∉ S) and (c_e < a[v]))
             decrease priority a[v] to c_e
}
```

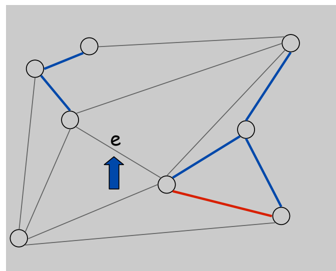
Special Case: Kruskal's Algorithm

Kruskal's algorithm. [Kruskal, 1956]

- Consider edges in ascending order of weight.
- Case 1: If both endpoints of e in same blue tree, color e red by applying red rule to unique cycle.
- Case 2: Otherwise color e blue by applying blue rule to cut consisting of all nodes in blue tree of one endpoint.



Case 1



Case 2

Implemention: Kruskal's Algorithm

Implementation. Use the union-find data structure.

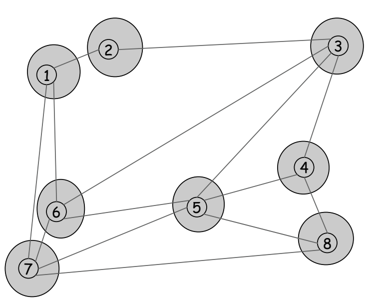
- Build set T of edges in the MST.
- Maintain set for each connected component.
- $O(m \log n)$ for sorting and $O(m\ \alpha\ (m, n))$ for union-find.

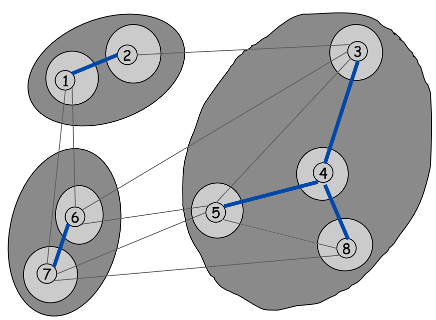
```
Kruskal(G, c) {
   Sort edges weights so that c1 ≤ c2 ≤ ... ≤ cm.
   T ← φ

   foreach (u ∈ V) make a set containing singleton u

   for i = 1 to m          are u and v in different connected components?
      (u,v) = ei
      if (u and v are in different sets) {
         T ← T ∪ {ei}
         merge the sets containing u and v
      }
   return T                merge two components
}
```


Special Case: Boruvka's Algorithm

Boruvka's algorithm. [Boruvka, 1926]

- Apply blue rule to cut corresponding to each blue tree.
- Color all selected edges blue.
- O(log n) phases since each phase halves total # nodes.

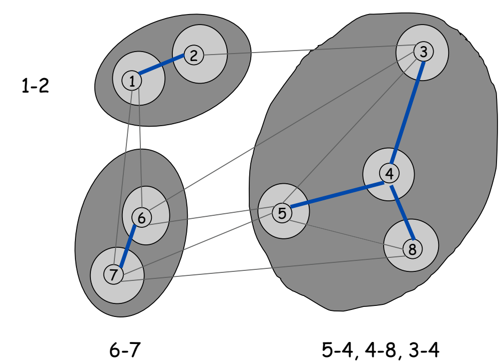




Implementing Boruvka's Algorithm

Boruvka implementation. $O(m \log n)$

- Contract blue trees, deleting loops and parallel edges.
- Remember which edges were contracted in each super-node.



MST Algorithms: Theory

Deterministic comparison based algorithms.

- $O(m \log n)$ Jarník, Prim, Dijkstra, Kruskal, Boruvka
- $O(m \log \log n)$. Cheriton-Tarjan (1976), Yao (1975)
- $O(m\, \beta(m, n))$. Fredman-Tarjan (1987)
- $O(m \log \beta(m, n))$. Gabow-Galil-Spencer-Tarjan (1986)
- $O(m\, \alpha(m, n))$. Chazelle (2000)

Holy grail. $O(m)$.

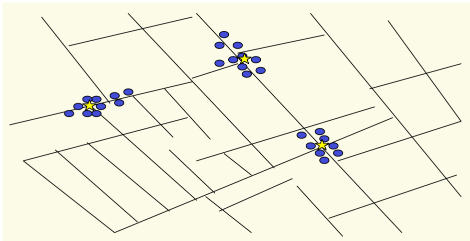
Notable.

- $O(m)$ randomized. Karger-Klein-Tarjan (1995)
- $O(m)$ verification. Dixon-Rauch-Tarjan (1992)

Euclidean.

- 2-d: $O(n \log n)$. compute MST of edges in Delaunay
- k-d: $O(k n^2)$. dense Prim

4.7 Clustering



Outbreak of cholera deaths in London in 1850s.
Reference: Nina Mishra, HP Labs

Clustering

Clustering. Given a set U of n objects labeled $p_1, \ldots, p_n$, classify into coherent groups.
(↑ photos, documents. micro-organisms)

Distance function. Numeric value specifying "closeness" of two objects.
(↑ number of corresponding pixels whose intensities differ by some threshold)

Fundamental problem. Divide into clusters so that points in different clusters are far apart.

- Similarity searching in medical image databases
- Skycat: cluster 2×10^9 sky objects into stars, quasars, galaxies.
- Routing in mobile ad hoc networks.
- Document categorization for web search.
- Identify patterns in gene expression.

Clustering of Maximum Spacing

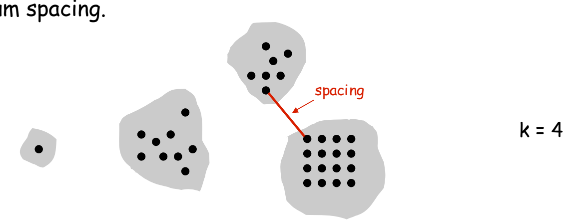
k-clustering. Divide objects into k non-empty groups.

Distance function. Assume it satisfies several natural properties.

- $d(p_i, p_j) = 0$ iff $p_i = p_j$ (identity of indiscernibles)
- $d(p_i, p_j) \geq 0$ (nonnegativity)
- $d(p_i, p_j) = d(p_j, p_i)$ (symmetry)

Spacing. Min distance between any pair of points in different clusters.

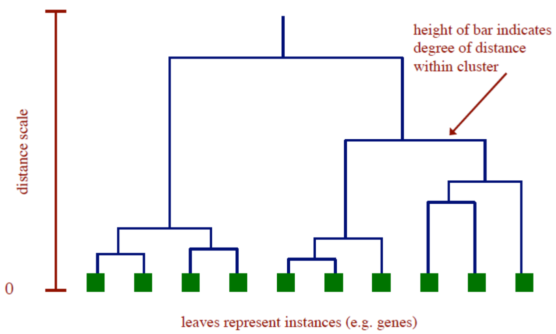
Clustering of maximum spacing. Given an integer k, find a k-clustering of maximum spacing.



Dendrogram

Dendrogram. Scientific visualization of hypothetical sequence of evolutionary events.

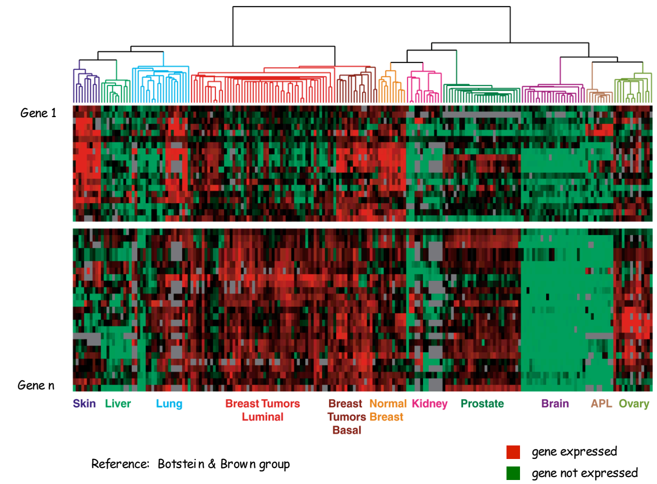
- Leaves = genes.
- Internal nodes = hypothetical ancestors.



Reference: http://www.biostat.wisc.edu/bmi576/fall-2003/lecture13.pdf

Dendrogram of Cancers in Human

Tumors in similar tissues cluster together.



Reference: Botstein & Brown group

Greedy Clustering Algorithm

Single-link k-clustering algorithm.

- Form a graph on the vertex set U, corresponding to n clusters.
- Find the closest pair of objects such that each object is in a different cluster, and add an edge between them.
- Repeat n-k times until there are exactly k clusters.

Key observation. This procedure is precisely Kruskal's algorithm (except we stop when there are k connected components).

Remark. Equivalent to finding an MST and deleting the k-1 most expensive edges.

Greedy Clustering Algorithm: Analysis

Theorem. Let C^* denote the clustering $C^*_1, \ldots, C^*_k$ formed by deleting the k-1 most expensive edges of a MST. C^* is a k-clustering of max spacing.

Pf. Let C denote some other clustering $C_1, \ldots, C_k$.

- The spacing of C^* is the length d^* of the (k-1)st most expensive edge.
- Let p_i, p_j be in the same cluster in C^*, say C^*_r, but different clusters in C, say C_s and C_t.
- Some edge (p, q) on p_i-p_j path in C^*_r spans two different clusters in C.
- All edges on p_i-p_j path have length $\le d^*$ since Kruskal chose them.
- Spacing of C is $\le d^*$ since p and q are in different clusters. ▪