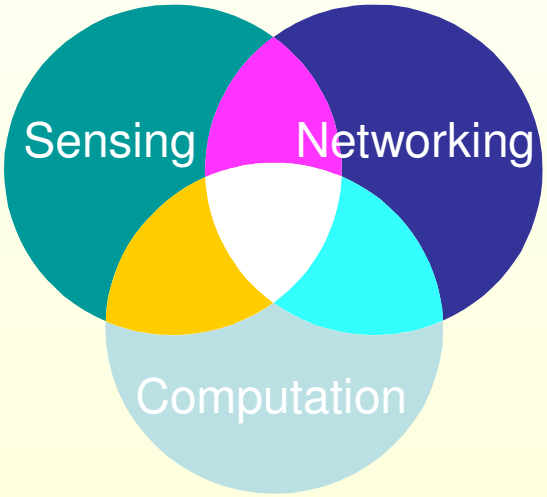
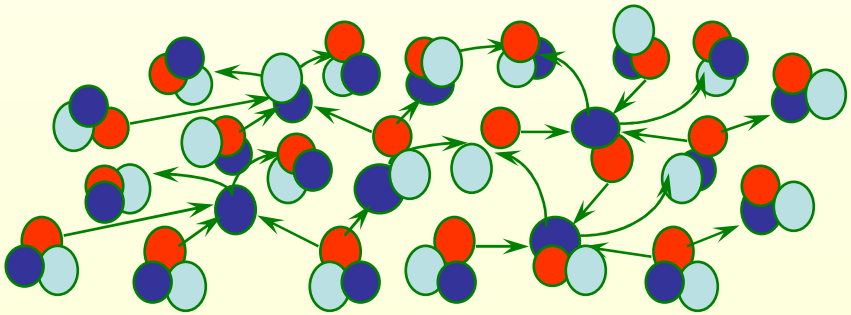
Networking Sensors, II



Leonidas Guibas
Stanford University



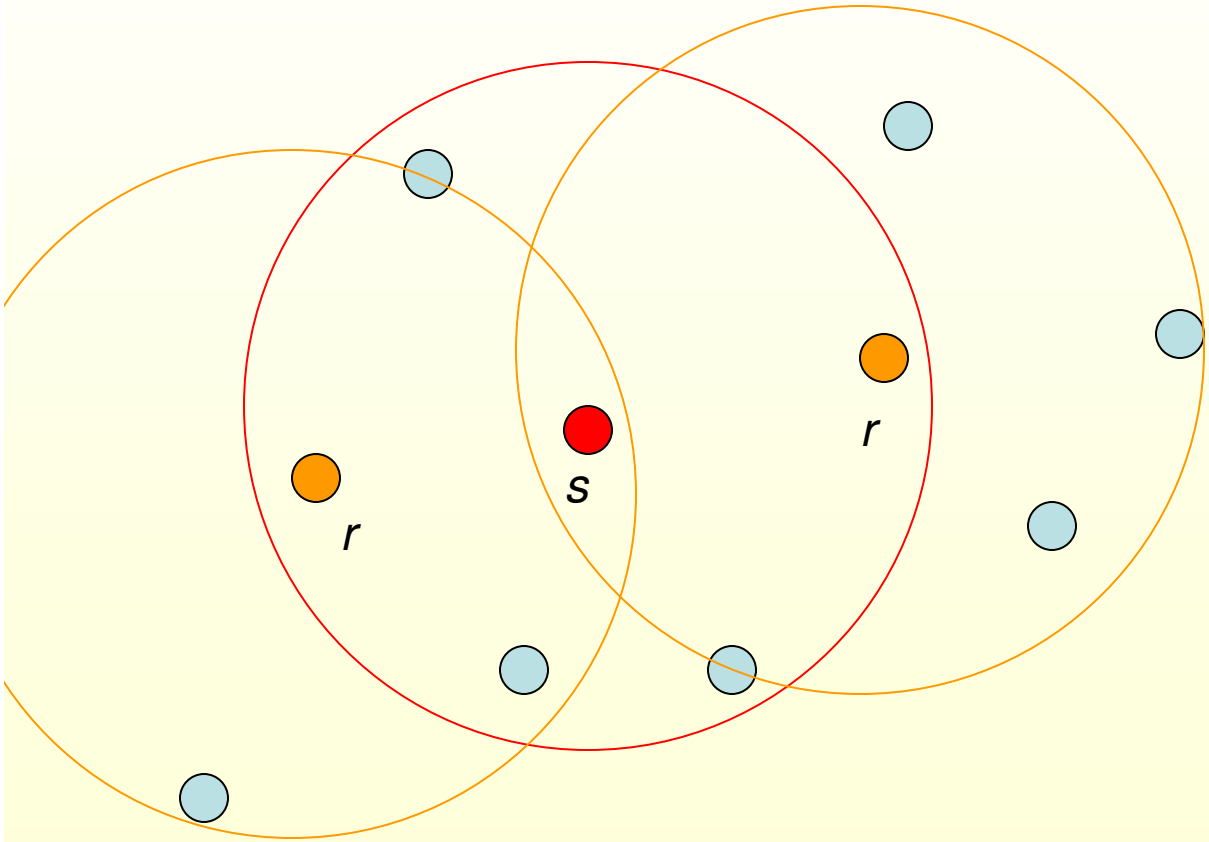
CS428

Beyond Point-to-Point Connections

Energy Minimizing Broadcast

- We have a network of n nodes with known positions
- A special node, the source s, wishes to transmit a packet so that it is received by all other nodes
- Intermediate nodes can act as relays for the packet
- Transmission power attenuates as $O(1/r^{\alpha})$
- The objective is to minimize the total energy used
- The result is realized by a optimal broadcast tree B

Energy Minimizing Broadcast



Relay solutions may be more efficient

The wireless multicast advantage

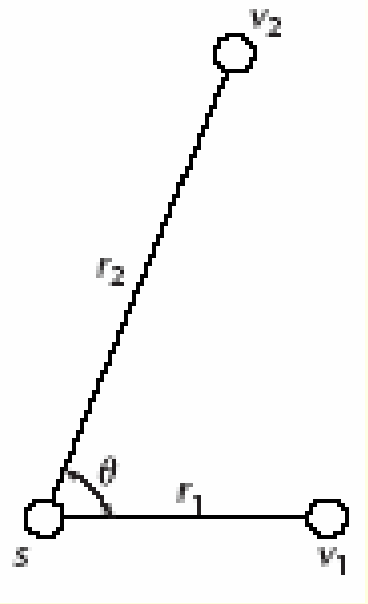
A Simple Example: Two Schedules

- A. *s* transmits, reaching both v_1 and v_2
- B. *s* transmits to reach only v_1, then v_1 transmits to reach v_2

$$(1 + x^2 - 2x\cos\theta)^{\alpha/2} > x^\alpha - 1\,,$$

- simplifies to

$$r_1 > r_2\cos\theta \qquad \text{if } \alpha=2$$



$$x = \frac{r_2}{r_1}$$

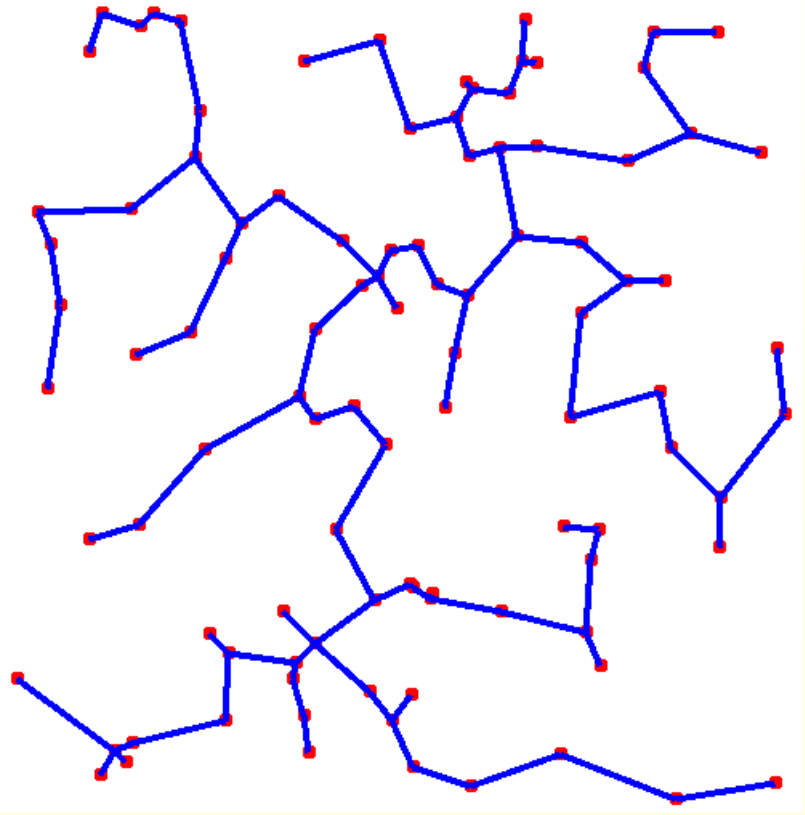
A Combinatorial Problem

- In the general case, the number of possible broadcast strategies is exponential in *n*; in fact, the minimum energy broadcast problem has been shown to be NP-complete
- Thus at best we can hope for an approximation algorithm, or a greedy algorithm that works well in practice

But, in the Wired Setting ...

- The same problem is classical and easy
- The Minimum Spanning Tree (MST) Problem:

Connect *n* nodes in pairs, so that all *n* nodes are connected together, and the total cost of all connections is minimized.



The MST is a Constant-Factor Approximation!

not so good, 6-12 ....

- The MST minimizes $\sum_e |e|^\alpha$
- Also, $\sum_{e \in \mathrm{MST}} |e|^\alpha = O(d^\alpha)$ for $\alpha \geq 2$

radius of a set of nodes

- For an optimal broadcast tree *B*

$$\sum_{e \in T_p} |e|^\alpha = O(r_p^\alpha), \qquad r_p^\alpha = \Omega(\sum_{e \in T_p} |e|^\alpha).$$

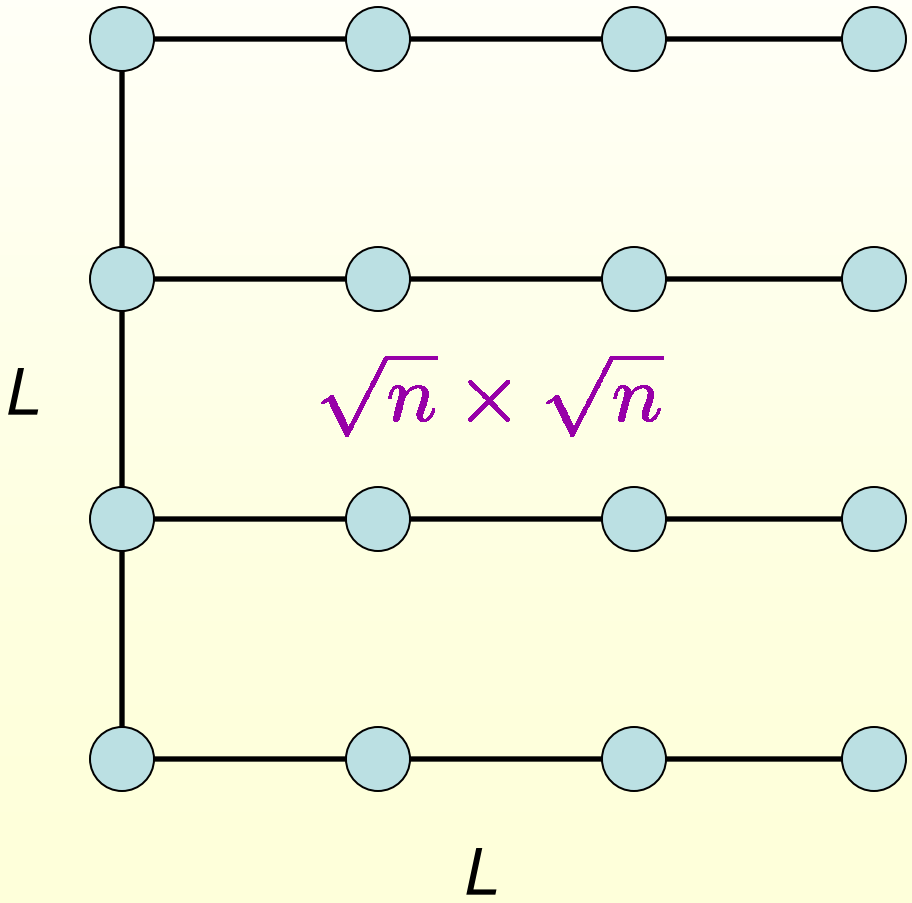
for relay node *p*, T_p is the MST of its children and itself, r_p its broadcast radius

$$\sum_p r_p^\alpha = \Omega(\sum_{e \in U} |e|^\alpha)$$

U, the union of the T_p's is a spanning tree of all nodes

$$(L/\sqrt{n})^{\alpha} n = L^{\alpha} n^{1-\alpha/2} = O(L^{\alpha})$$

if $\alpha \geq 2$

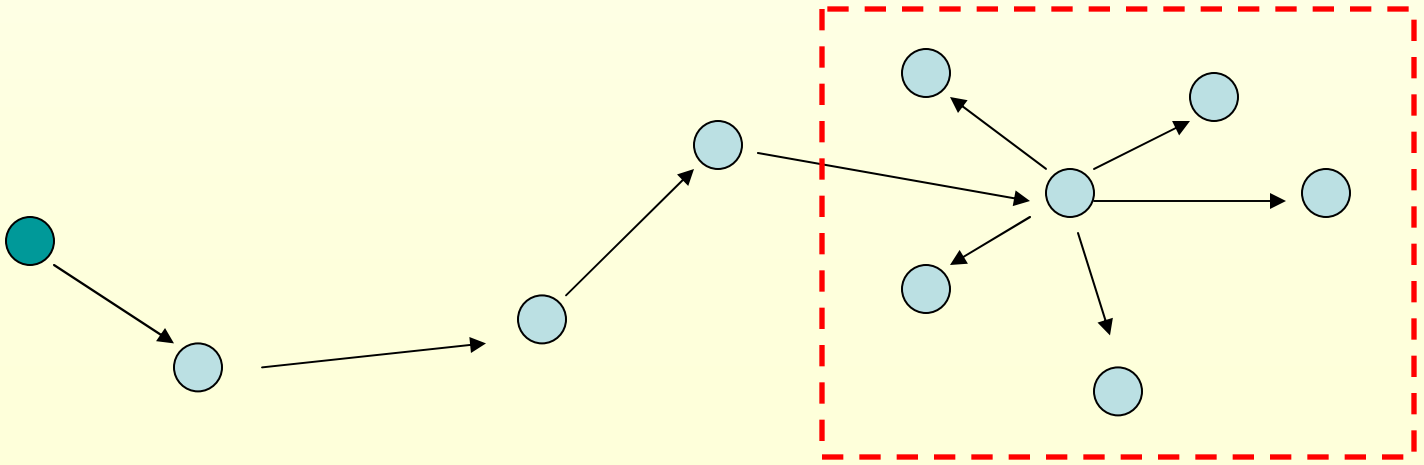


Broadcast Incremental Power (BIP) Algorithm

- Like in the Prim-Jarnik MST algorithm, add nodes one at a time to the tree
- At each step, choose the node that has the minimal incremental cost to be connected to one of the existing nodes
- No theoretical guarantees, but works well in practice

Energy-Aware Routing to a Region

- Broadcast a packet to all nodes within a geographic region
  - get the packet to the region
  - distribute the packet within the region
  - minimize energy use



Learning Real-Time A* Algorithm

- Choose a destination *d* within the desired region *R*
- Each node *y* maintains a learned cost *h*(*y*,*d*) to the destination *d*
- Initially *h*(*y*,*d*) is set as follows:

$$h(y,d) = \alpha \ell(yd) + (1-\alpha) E_y \,,$$

straight-line distance to *d*

energy left at *y*

balance route cost and energy reserves

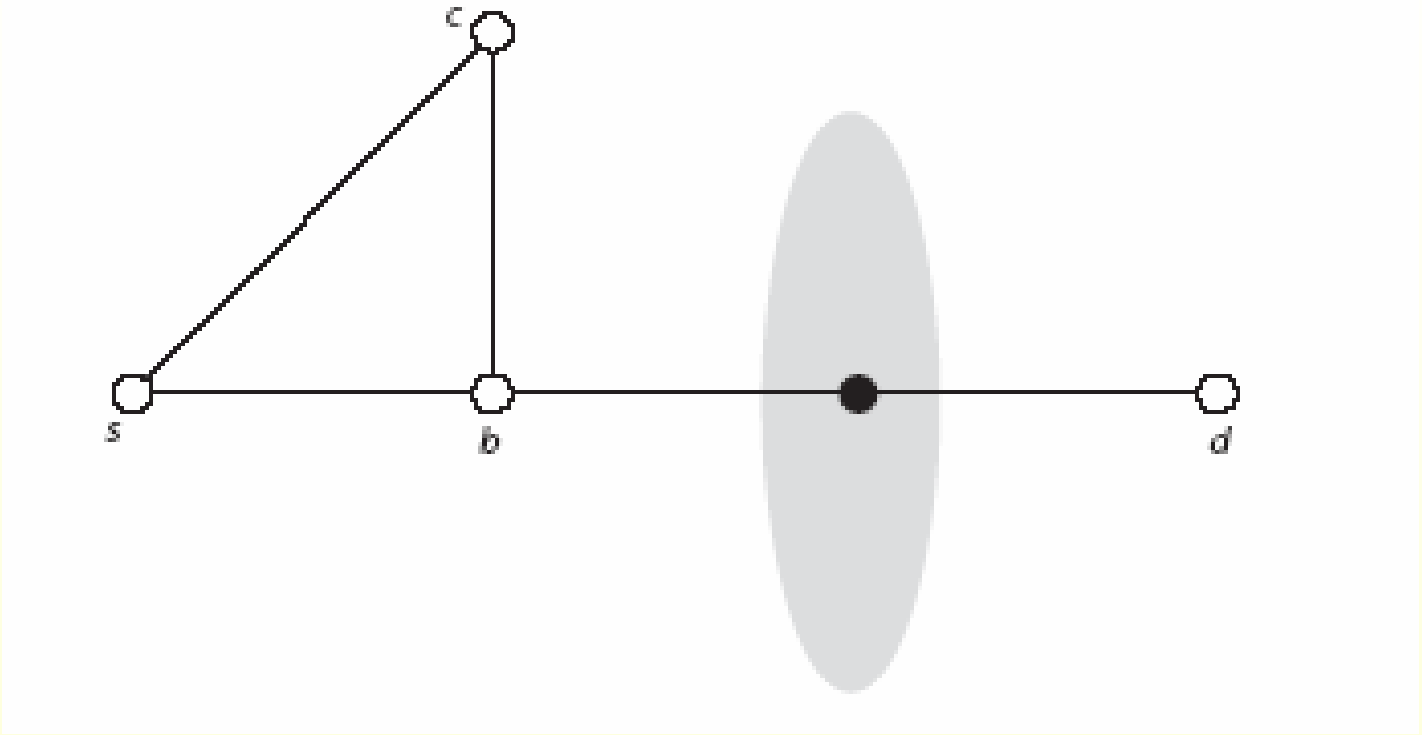
The GEAR Routing Protocol

- Greedy neighbor selection:
  - if neighbors exist closer to the destination (Euclidean sense), choose the one of smallest learned cost
  - otherwise choose the neighbor of smallest learned cost (among all neighbors)
- Relaxation: periodically broadcast learned cost to neighbors: *y* sends to *x*

$$h(x,d) \leftarrow \min\{c(x,y) + h(y,d), h(x,d)\}\,,$$

where $c(x,y) = \alpha\ell(xy) + (1-\alpha/2)E_x + (1-\alpha/2)E_y$

An Example



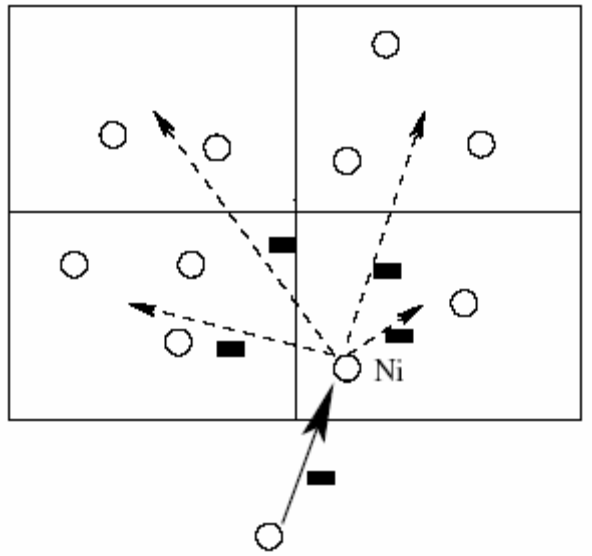
Initially *s* chooses *b* over *c*, but in time learns that *c* is actually a better choice

LRTA* Properties

- The algorithm will eventually converge to a a locally minimal path
- Convergence can be slow (quadratic in the number of nodes)
- Behaves reasonably well in practice
- Leads to better network lifetimes that GPSR

Distribution within the Region R

- Recursive subdivision and forwarding



- Restricted flooding
  - can be wasteful

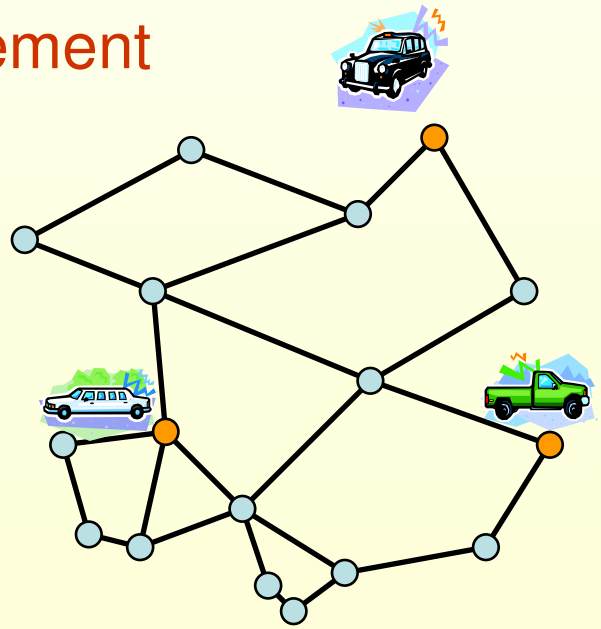
Steiner-Tree Problems

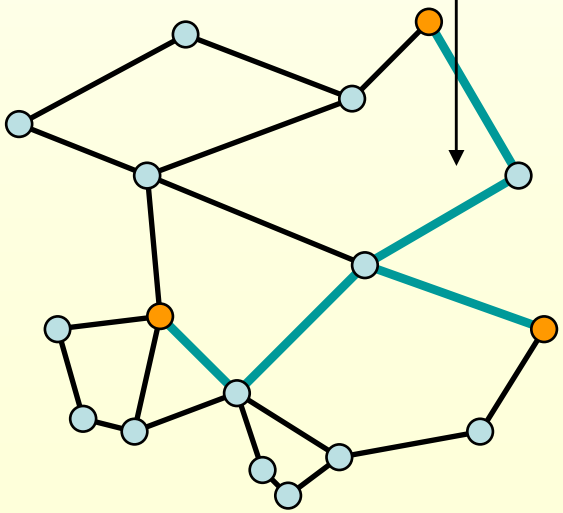
Connectivity Among Agents Tracking Mobile Targets

In a sensor network, we wish to maintain a communication subnetwork among processes tracking mobile targets.

Key component of the group management problem.

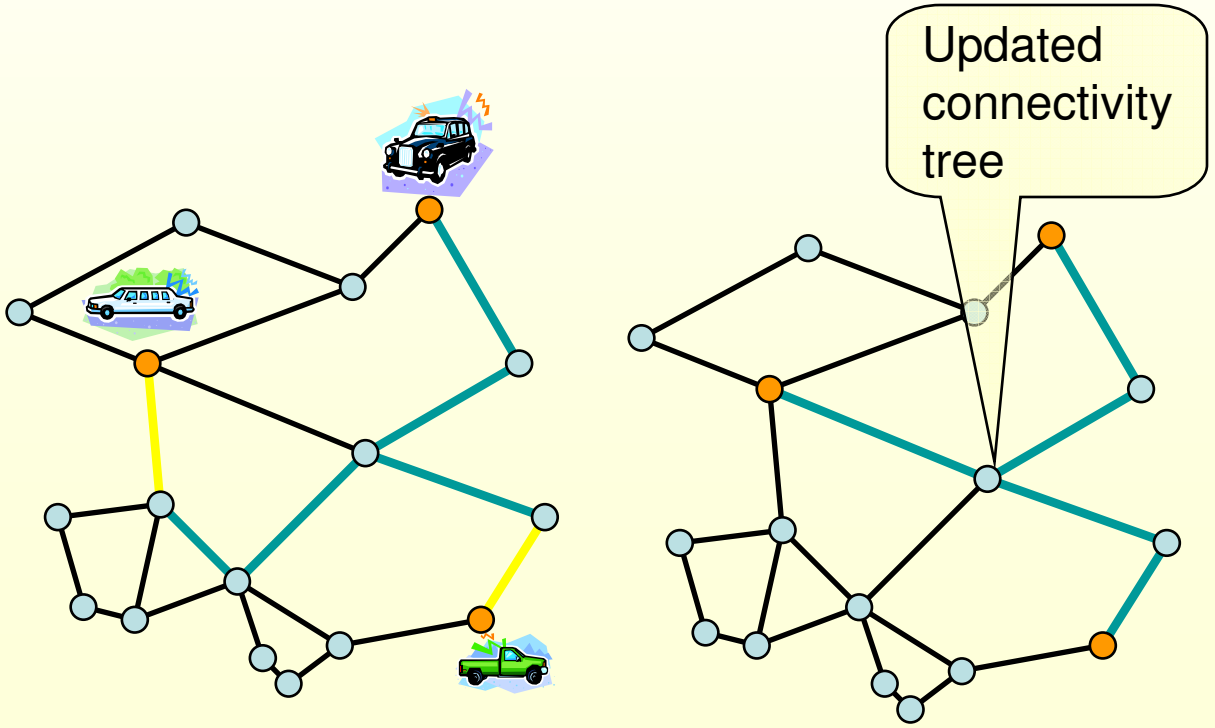
Agents, or processes, hop from node to node.





Evolving the Tree as Targets Move

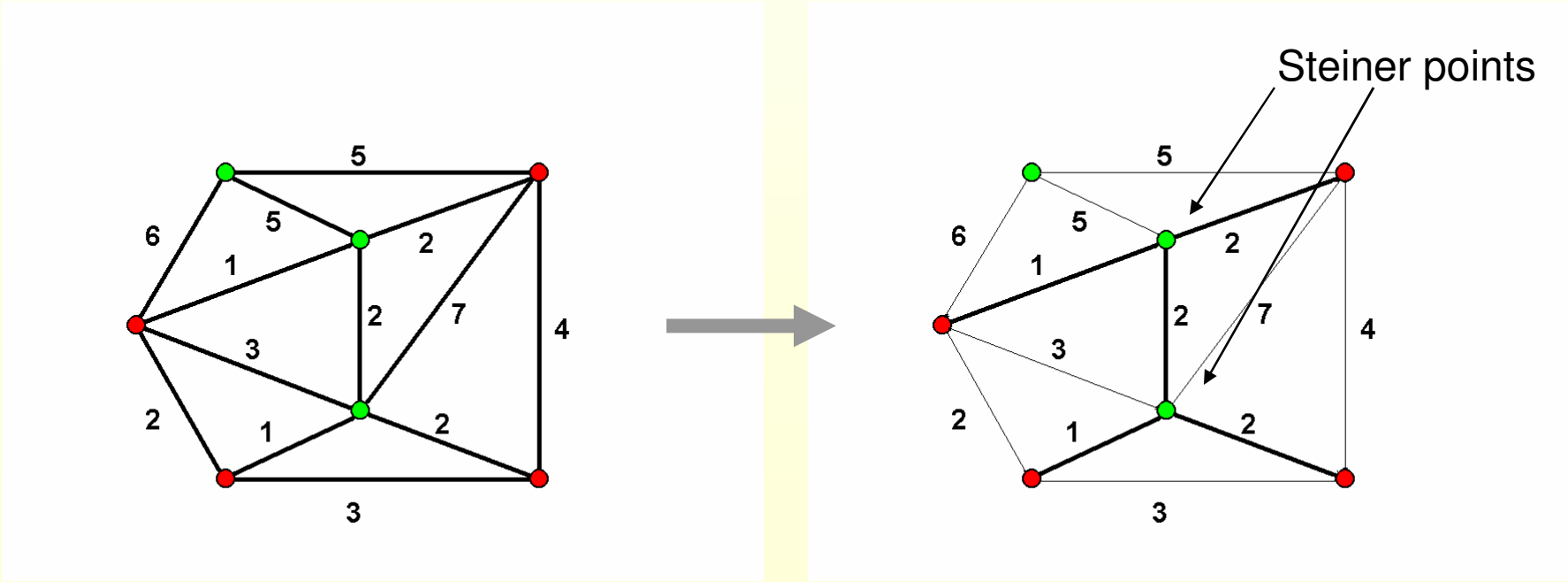
We study techniques for updating the communication network tree, as agents hop between nodes while following the targets.



Useful for
- identity management
- multi-target tracking
- collaborative exploration

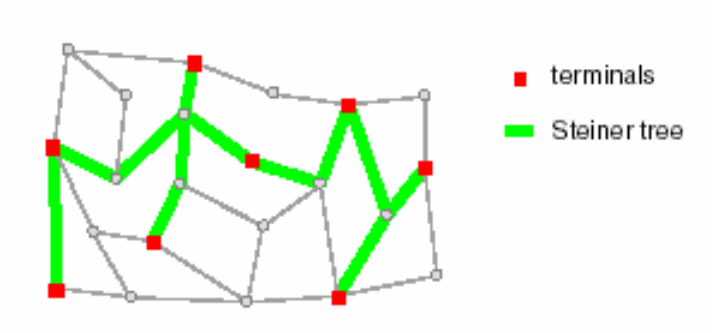
The Minimum Steiner Tree Problem

- Find the least total weight subtree connecting a particular subset of the nodes of $G = (V,E)$ (the terminals K) --- additional branching (Steiner) points are permitted



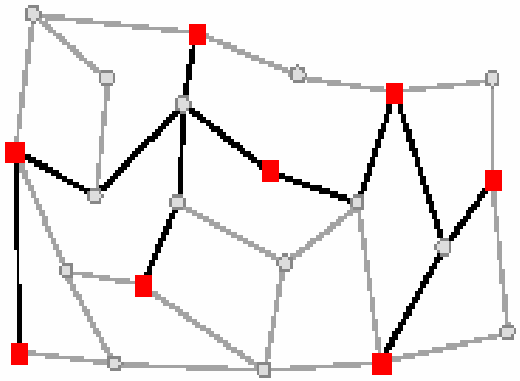
Minimum Steiner Tree Problem Facts

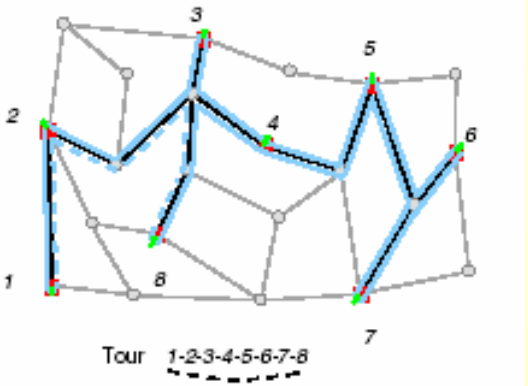
- Named after Jacob Steiner (1796-1863)
- When $K = V$, it is the MST problem
- When $|K| = 2$, it is the shortest path problem
- But unlike these two problems, solving the general case of the ST problem is NP-complete
- Can be solved in $O(n^k)$ time by exhaustive enumeration of Steiner points ($n = |G|$, $k = |K|$)
- Focus on polynomial-time approximation algorithms



Using the MST for Approximation Ratio ρ = 2

- Consider a graph whose nodes are the terminals K and edges between them have weights equal to the corresponding shortest paths in G
- Compute an MST of this new graph
- Embed it in the original graph, drop duplicate edges





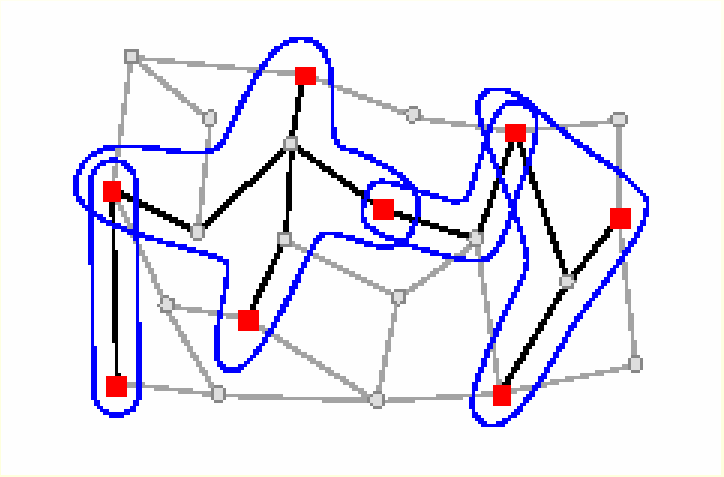
Same Idea, Extended to Hyper-graphs

- Define hyper-edges by considering all optimal Steiner subtrees based on *r* of the *k* terminals
- Compute MST on the hyper-graph
- Can get [Zelikovsky, ...]

$$\rho_2 = 2$$

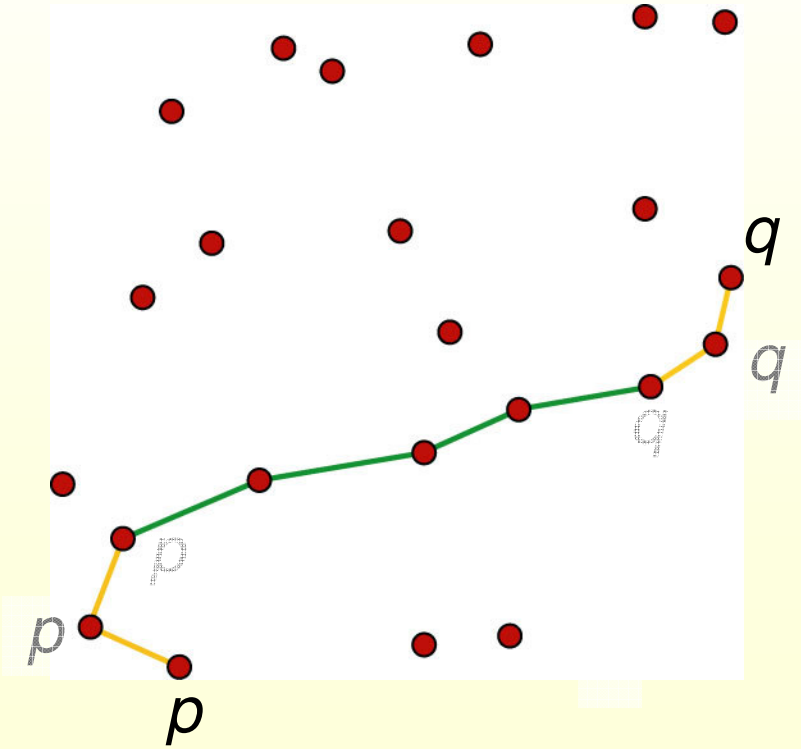
$$\rho_3 = 5/3$$

$$\rho_r \to 1, r \to \infty$$



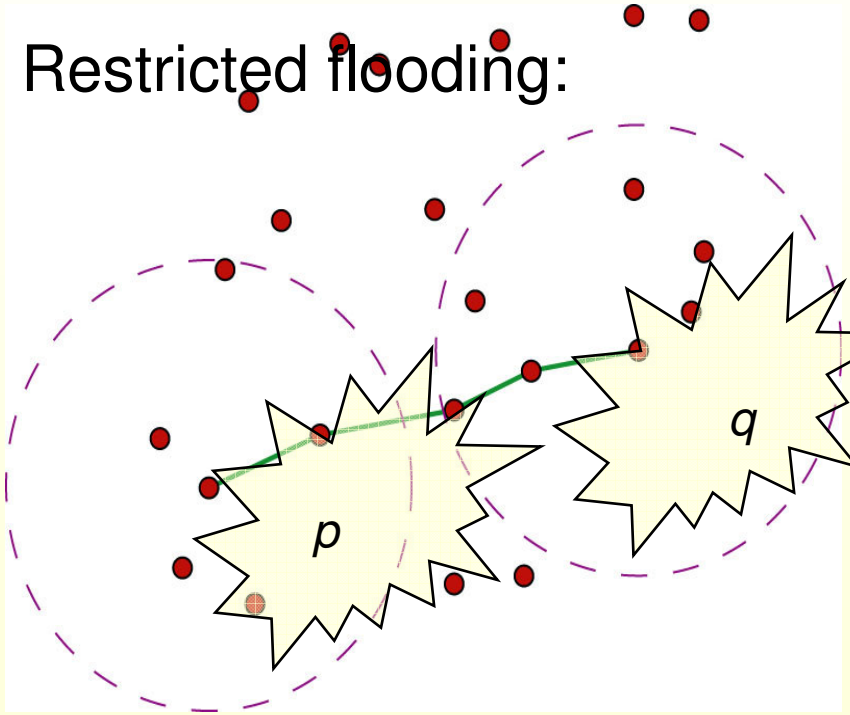
Amortized Path Updates for Terminal Motion

- Assume we have a good path between *p* and *q*.
- How can we maintain this path as *p* and *q* hop around?

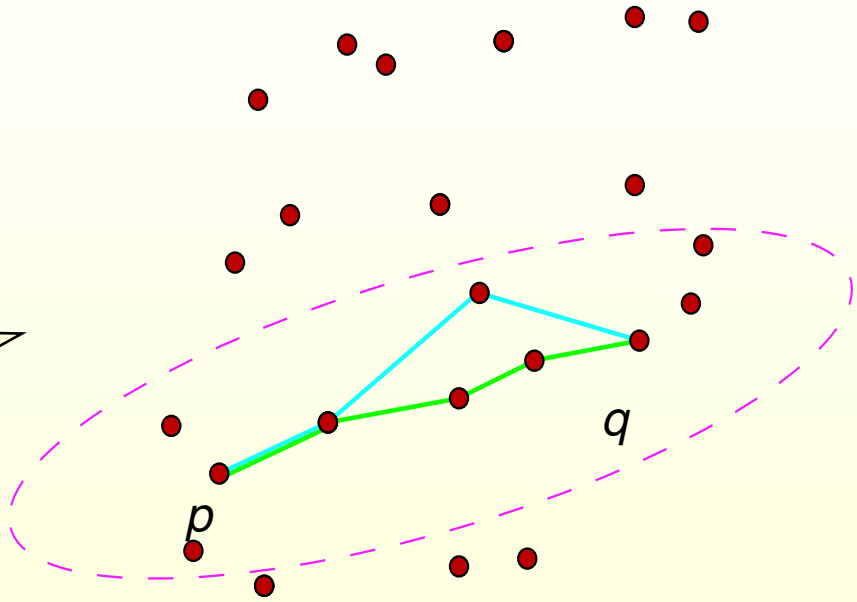


The old path can still be used, as long as *p* and *q* move only by a fraction of its length

Delay-and-Repair: Every so often, Re-compute a Good Path from *p* to *q*



Wavefront (contour) propagation



Search can be confined inside the ellipse defined by the current path

Amortized Computation Cost, no Preprocessing

- If node density is constant and the path length of p to q is d, then the shortest path process visits $O(d^2)$ nodes, of which $O(d)$ are active at any one time (wavefront plus path nodes).
- This happens every $\Omega(d)$ steps (agent hops), and thus the amortized cost per hop is $O(d)$ – which is best possible up to a constant factor.

Extension to the Multi-Terminal Case

- Case of one mobile agent: maintain an approximate shortest path tree to the rest
  - bucket other terminals into groups according to agent distance: [0,1], [1,2], [2,4], [4,8], ...
  - use previous delay-and-repair approach at a frequency appropriate for each group
- Cumbersome, but extends to the all-mobile terminal case
- Still gives the same $O(d)$ amortized bound per agent hop

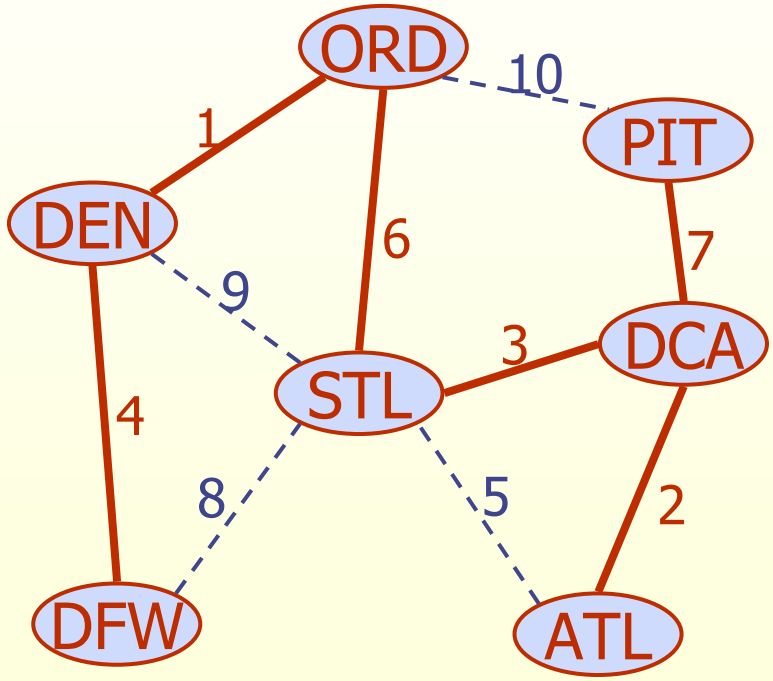
The End

Review of Some Classical Graph Algorithms

Minimum Spanning Trees

Minimum spanning tree (MST)

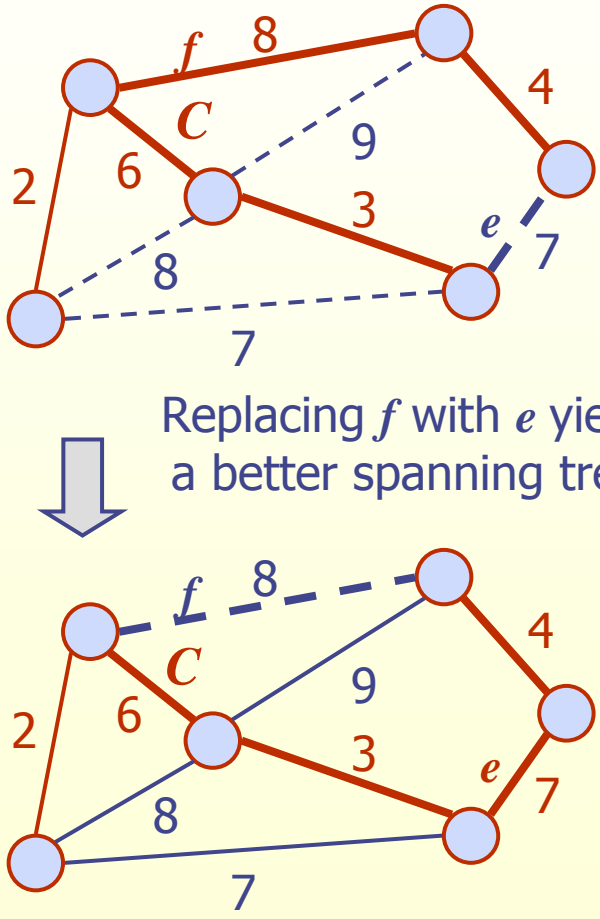
- Spanning tree of a weighted graph with minimum total edge weight
- Classic graph optimization problem
- Can be solved fast by a variety of greedy algorithms



Key MST Properties: Cycle Property

Cycle Property:

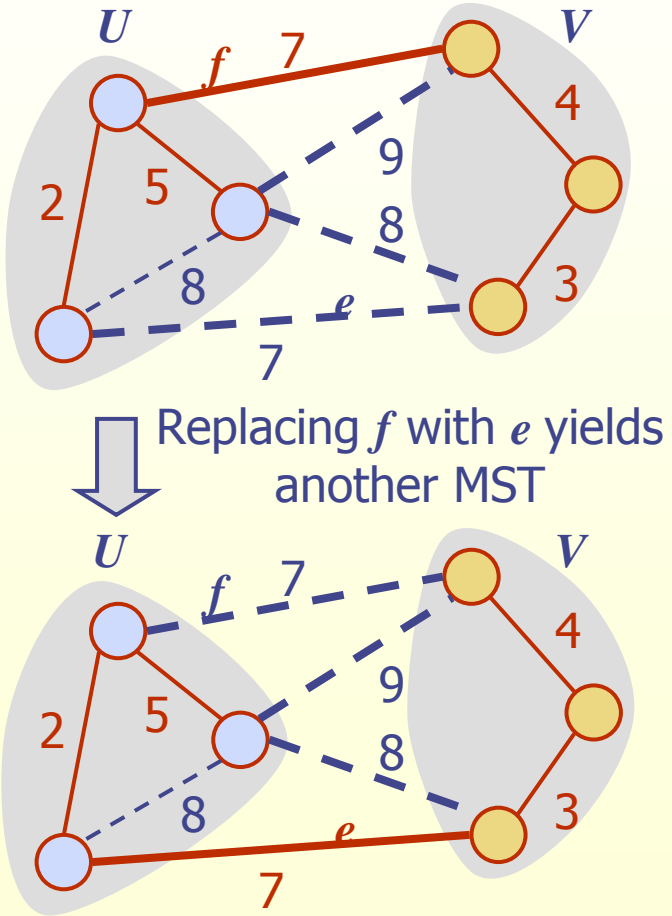
- Let T be a minimum spanning tree of a weighted graph G
- Let e be an edge of G that is not in T and C let be the cycle formed by e with T
- For every edge f of C, $weight(f) \leq weight(e)$



Key MST Properties: Partition Property

Partition Property:

- Consider a partition of the vertices of G into subsets U and V
- Let e be an edge of minimum weight across the partition
- There is a minimum spanning tree of G containing edge e

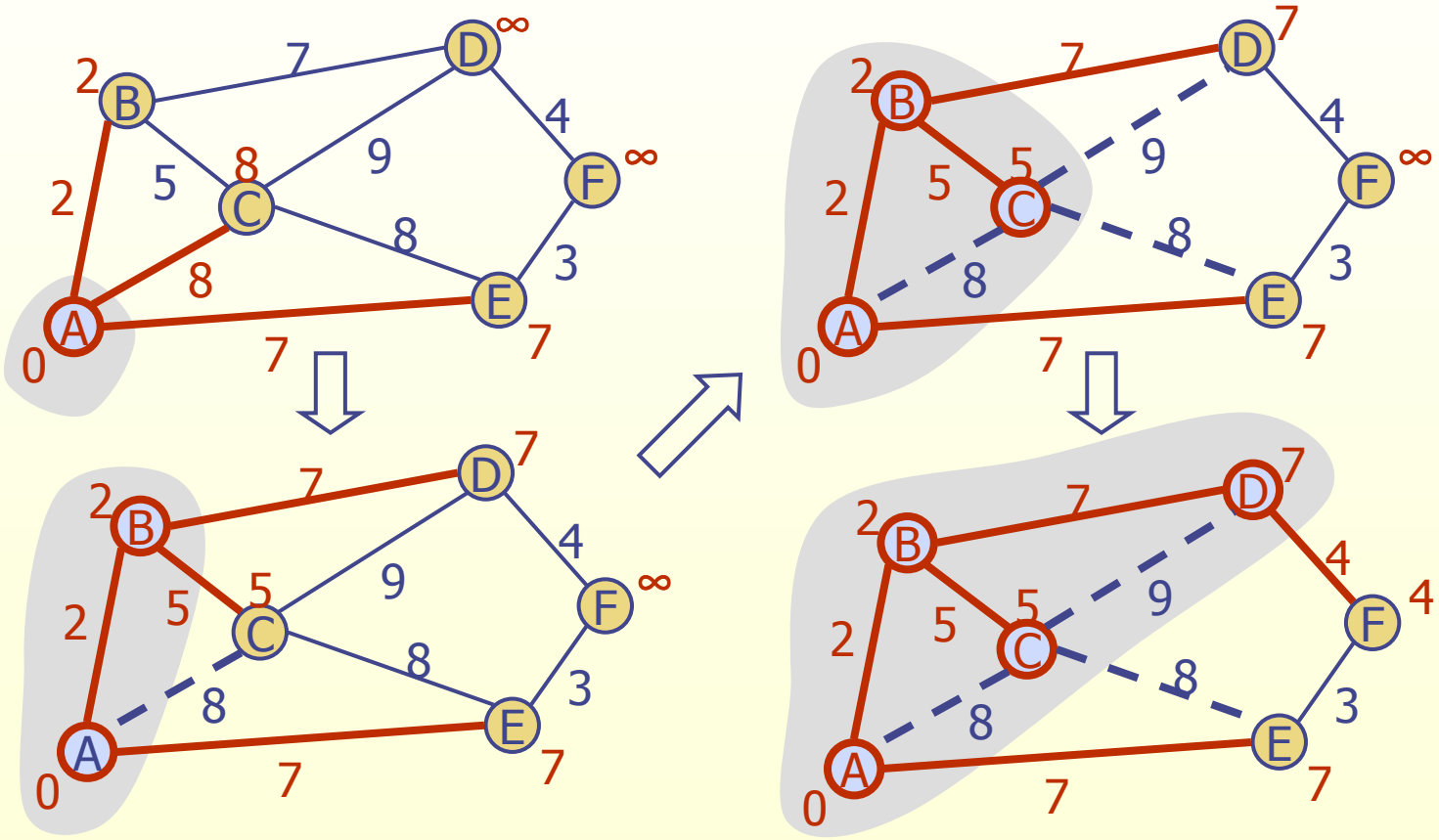


Famous MST Algorithms

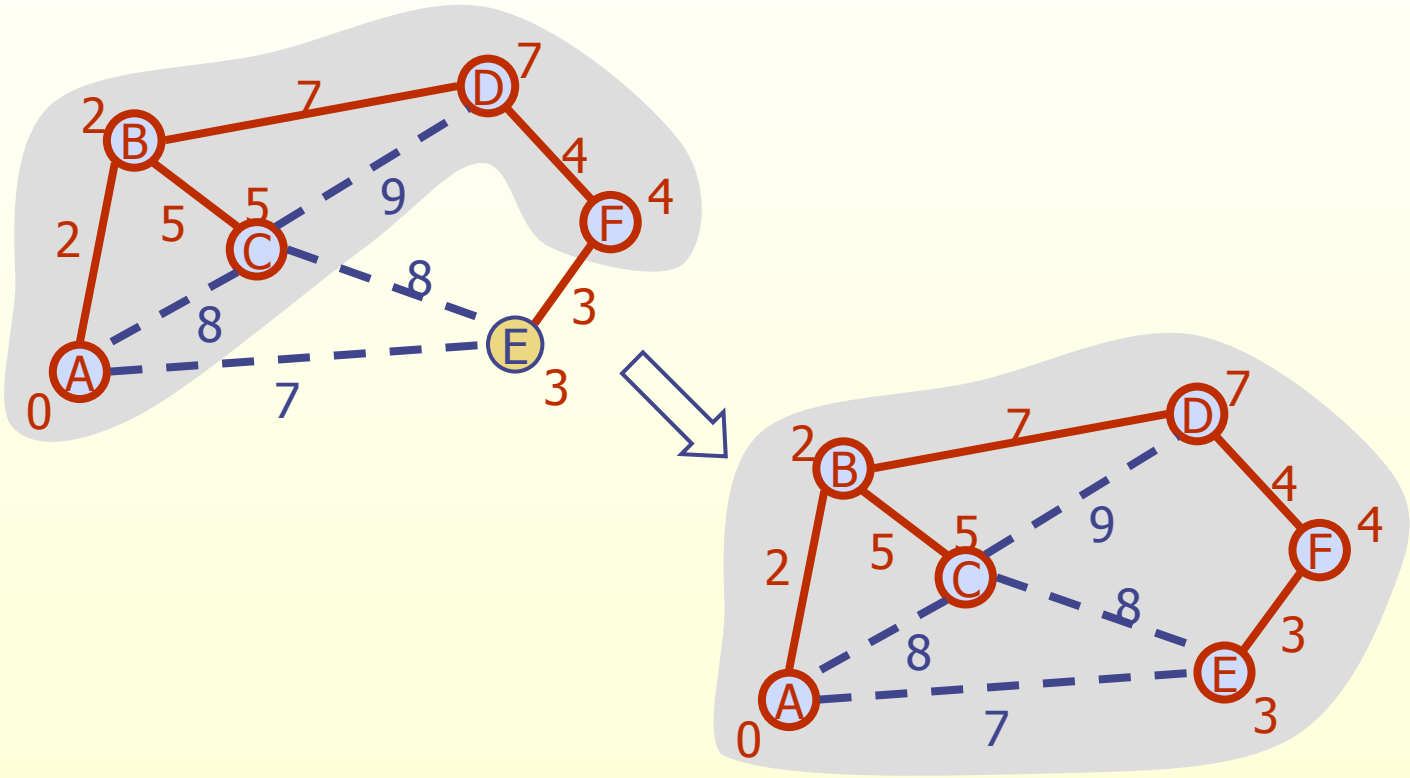
- *Prim-Jarnik*
  - Like Dijkstra's, grows single component MST
- *Kruskal*
  - inserts edges in weight-sorted order
- *Boruvka*
  - grows mutliple chunks of MST in parallel

- All work by making greedy choices
- All have complexity roughly linear in the number of vertices and edges of the underlying graph G
- Boruvka's can be adapted for distributed implementations (Gallager *et. al.)*

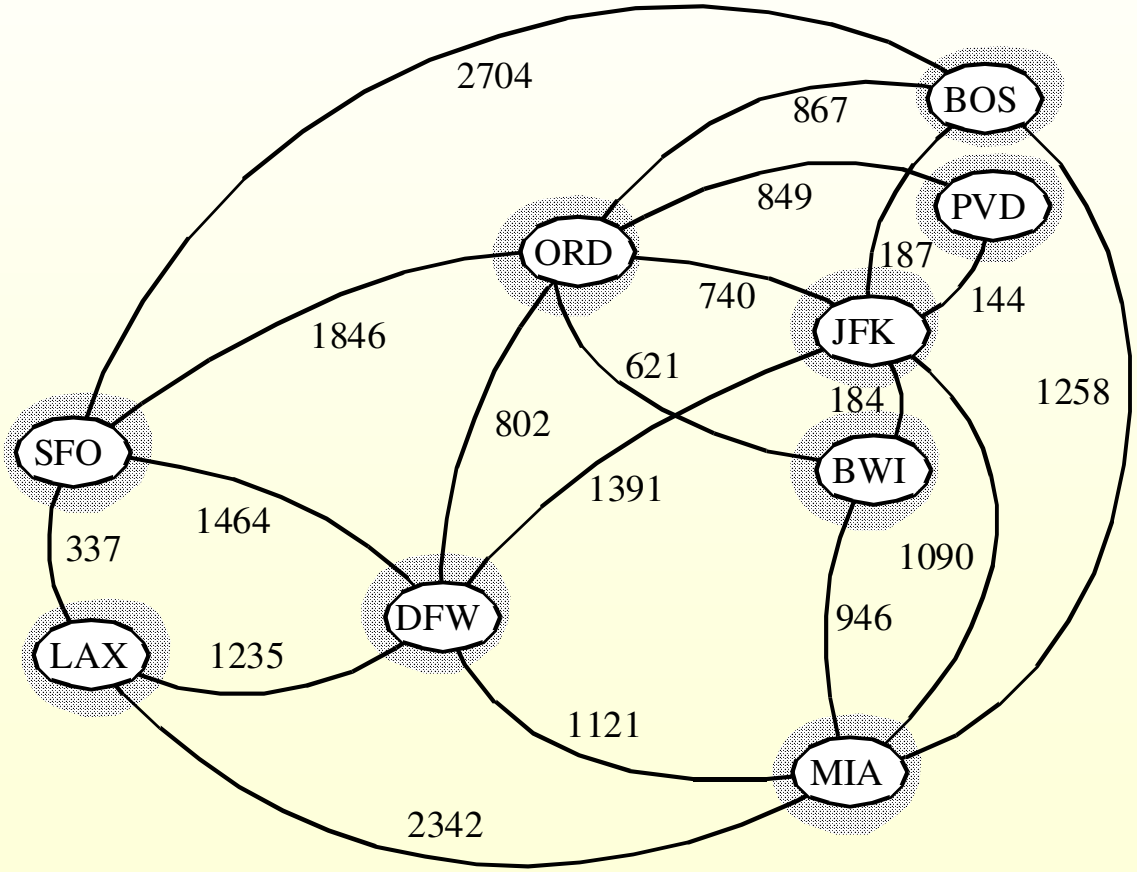
Prim-Jarnik in Action, I



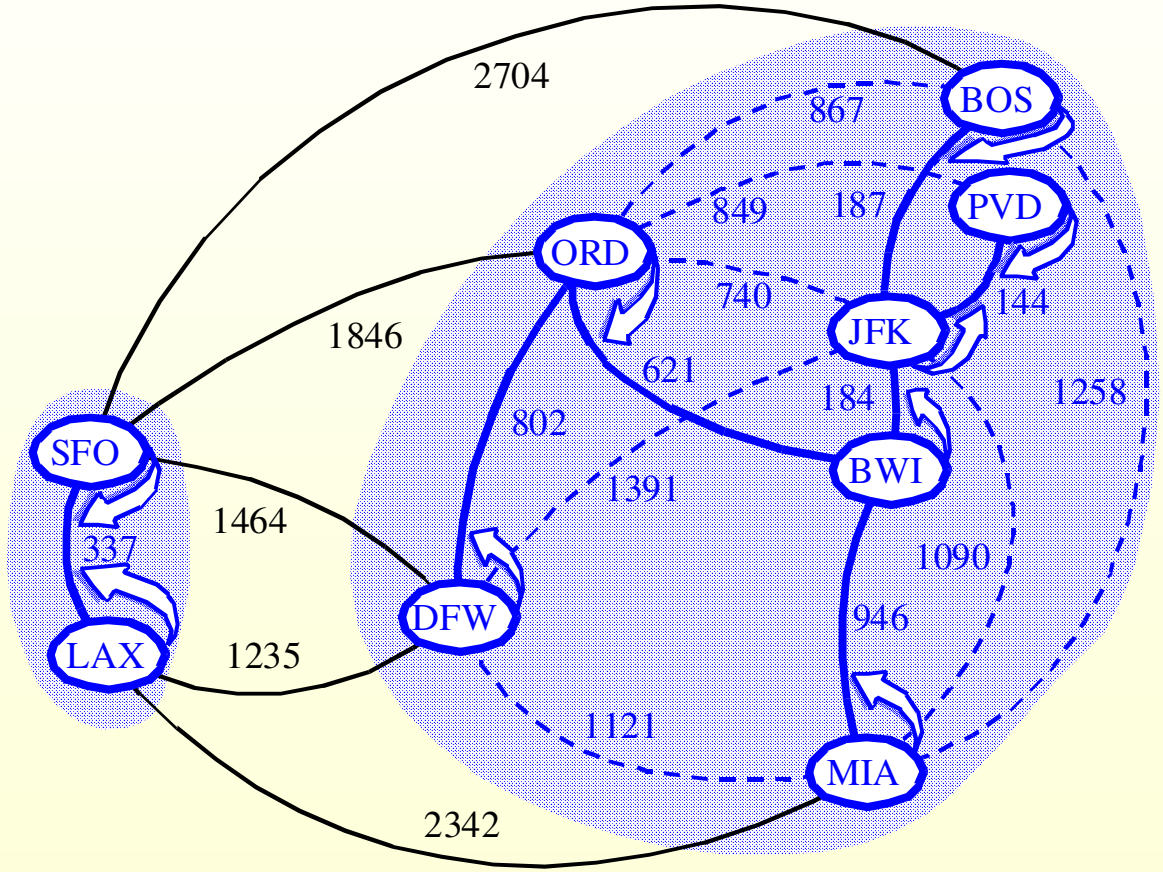
Prim-Jarnik in Action, II



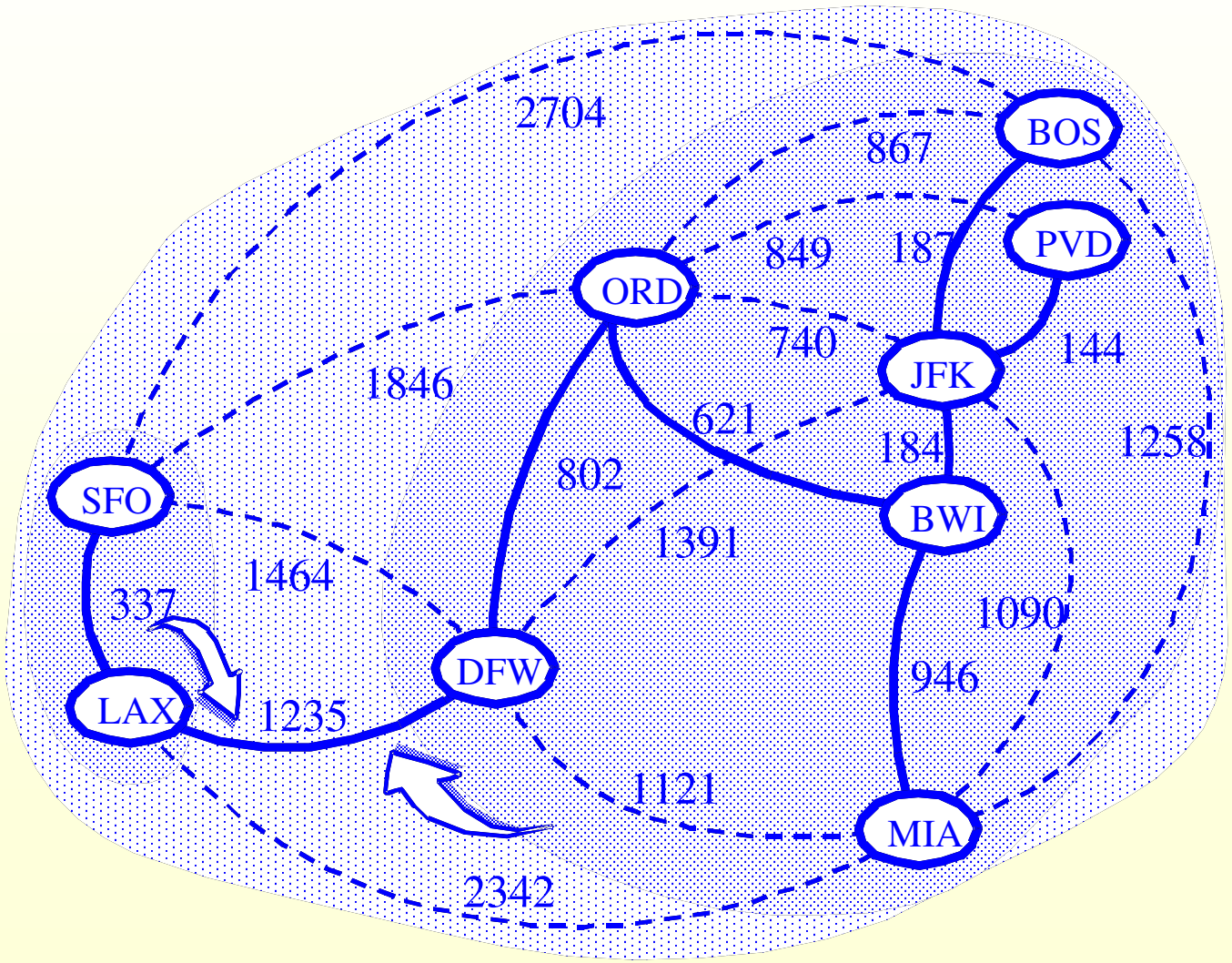
Boruvka in Action, I



Boruvka in Action, II



Boruvka in Action III



More Useful Properties of MSTs

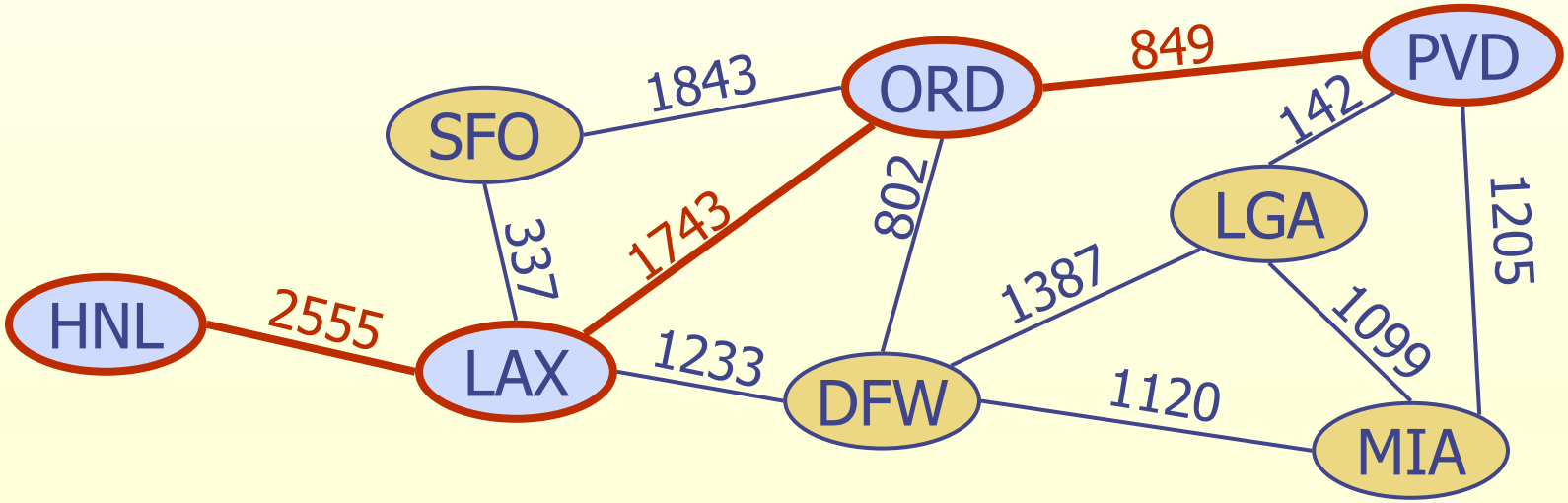
- If all edge weights are distinct, the MST is unique
- The MST topology only depends on edge weight comparisons
  - Edge weights of the form

$$\text{length}(e)^{\alpha}$$

for $\alpha > 0$ all give rise to the same MSTs

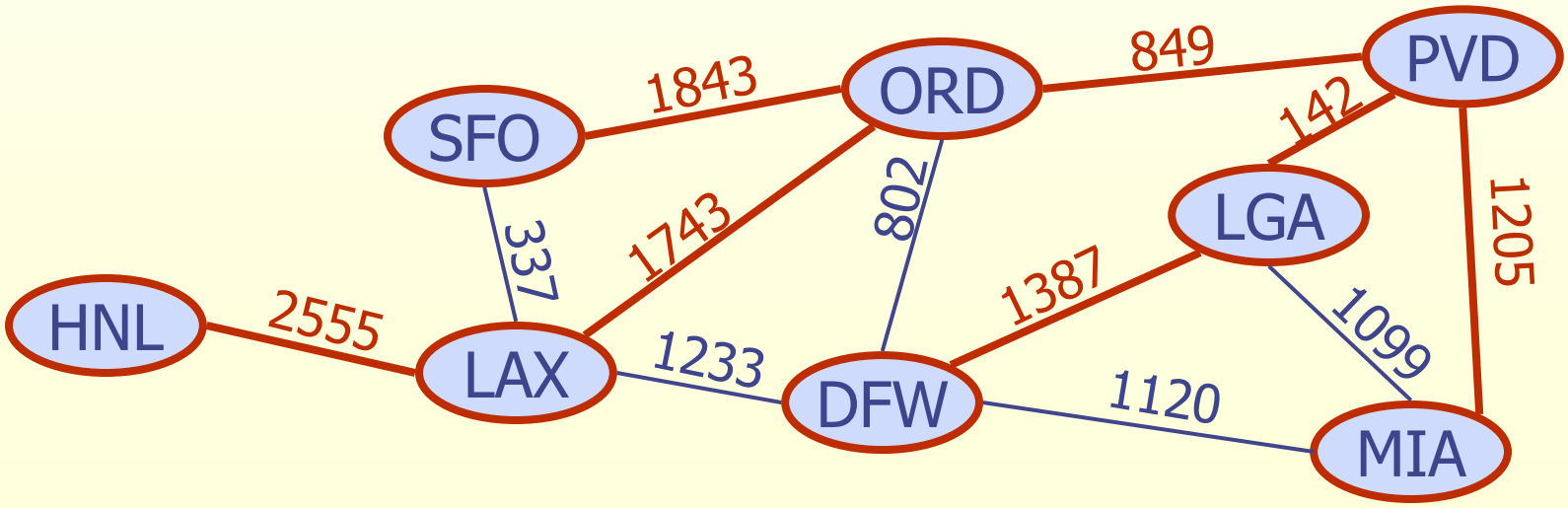
Shortest Path Problems: Single Pair

- Given a weighted graph and two vertices u and v, we want to find a path of minimum total weight between u and v (PVD to HNL)



Shortest Path Problems: Single Source

- Find a tree of shortest paths from one vertex to all others (PVD to all)

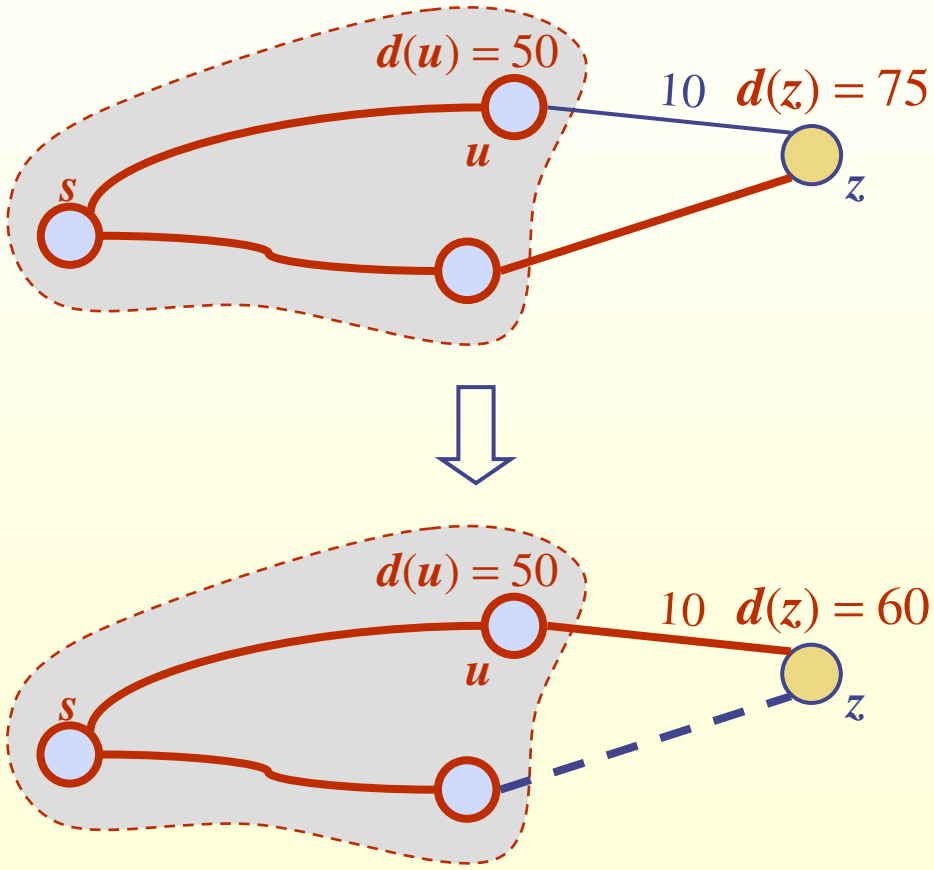


Key Facts About Shortest Paths

- A subpath of a shortest path is itself a shortest path
- Again, many famous efficient algorithms
  - Bellman-Ford
  - Dijkstra
  - Johnson
  - Floyd-Warshall
- Complexity of single pair same as that of single source

Edge Relaxation Step

- Relaxing the edge uz
- Improves distance estimate to node d
- Algorithms differ in the order in which relaxation steps are applied



Back to the Minimum Energy Broadcast