CSE 421
Introduction to Algorithms

Lecture 25

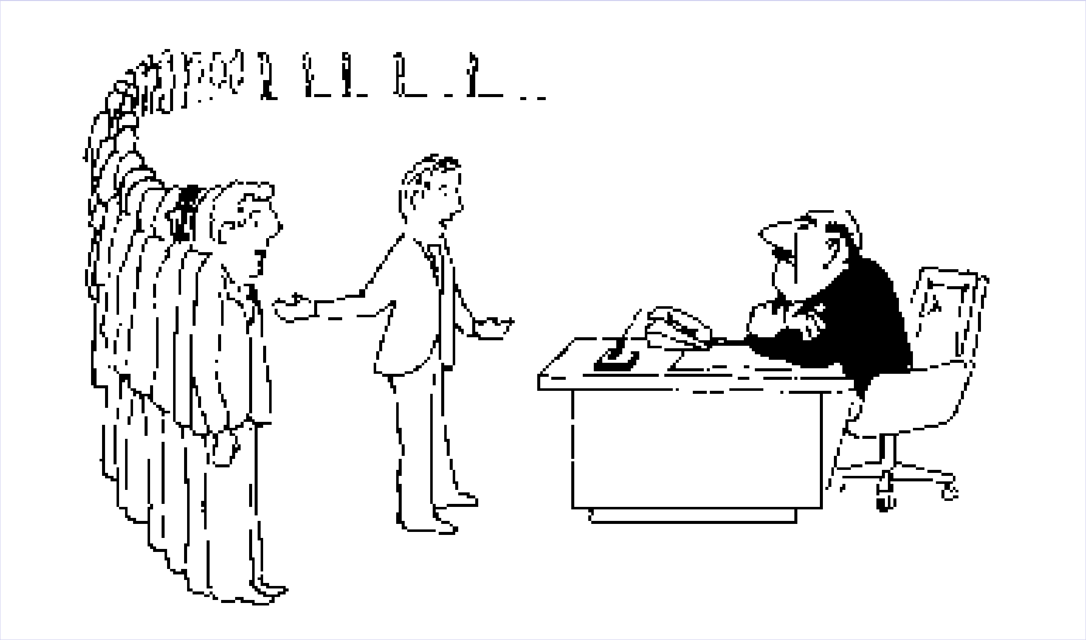
Coping with NP-Completeness

Announcements

- Today, Coping with NP-Completeness
  - Chapters 11 and 12
- Friday, Beyond NP-Completeness
  - Section 8.9, Chapter 9
- Homework 9, Due Friday, March 8
- Final exam,
  - Monday, March 11, 2:30-4:20 pm PDT
  - Comprehensive (~60% post midterm, ~40% pre midterm)
  - Old finals / answers on home page

Coping with NP-Completeness

- Approximation Algorithms
- Exact solution via Branch and Bound
- Local Search



I can’t find an efficient algorithm, but neither can all these famous people.

Approximation Algorithms

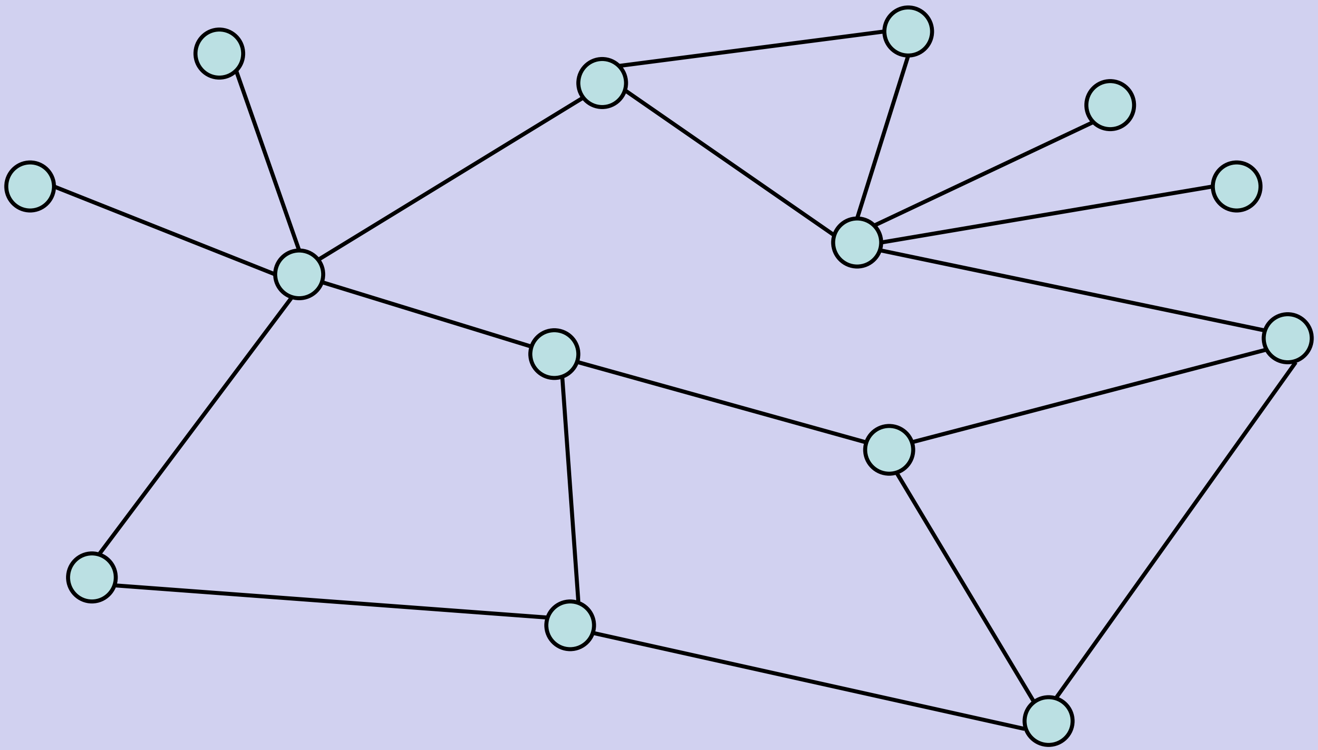
- K-Approximation Algorithm
- Worst case ratio of solution and optimal as input size goes to infinity
- Minimization problems
  - Find a solution at most K times the optimum
- Maximization problems
  - Find a solution at most 1/K times the optimum

Vertex Cover

- A vertex cover is a subset of the vertices that is adjacent to every edge
- VC is NP-Complete

```
W = {};
E′= E
while E′ is not empty
   Select e = (u,v) from E′
   Add u and v to W
   Remove all edges adjacent to u or v from E′
```

```
W = {};
E′= E
while E′ is not empty
    Select e = (u,v) from E′
    Add u and v to W
    Remove all edges adjacent to u or v from E′
```

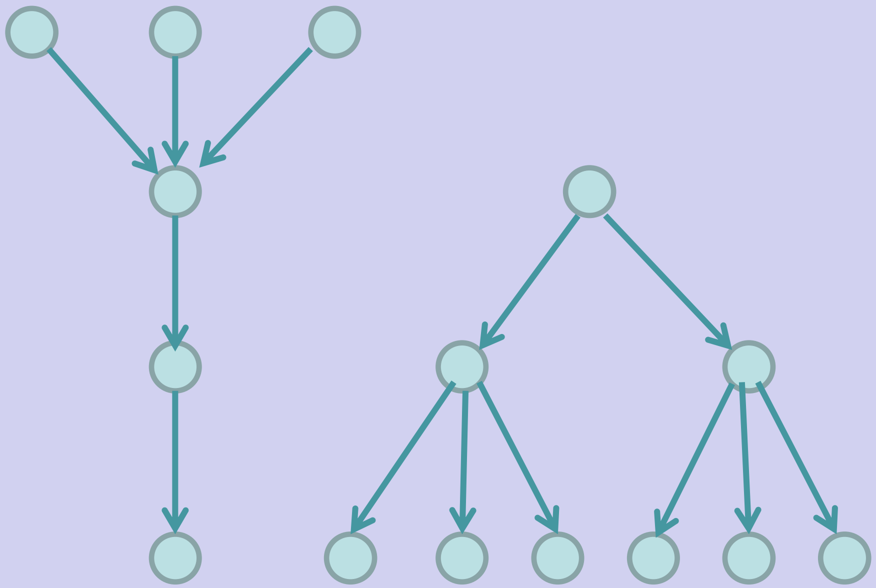


VC 2-Opt Bound

- When edge e = (u,v) is selected, neither u nor v is in W
- At least one of u or v must be in the VC to cover e
- Thus, at least ½ the vertices placed in W are necessary

Multiprocessor Scheduling

- Unit execution tasks
- Precedence graph
- K-Processors



- Polynomial time for k=2
- Open for k = constant
- NP-complete if k is part of the problem

Highest level first is 2-Optimal

Choose k items on the highest level

Claim: number of rounds is at least twice the optimal.

Suppose the maximum height of a task is H

A partial round removes < k elements

A full round removes k elements

2-Opt Proof for HLF

The number of partial rounds is at most H

Opt $\geq$ H

The number of full rounds is at most N / k

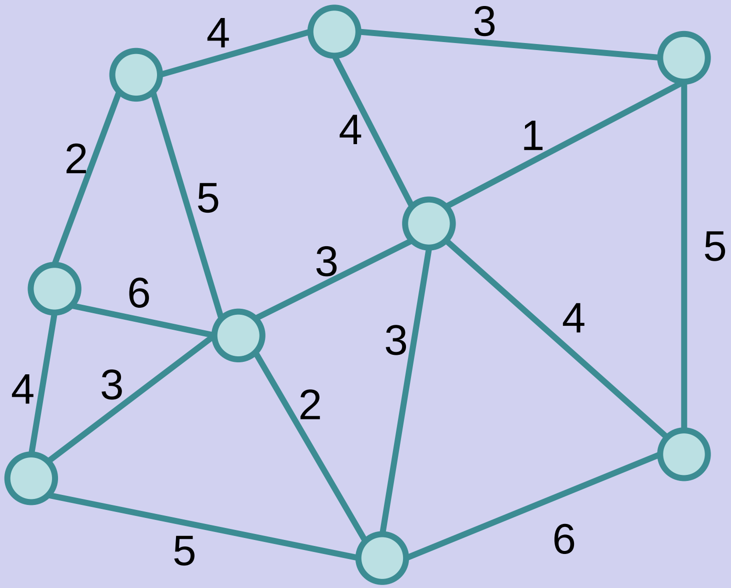
Opt $\geq$ N / k

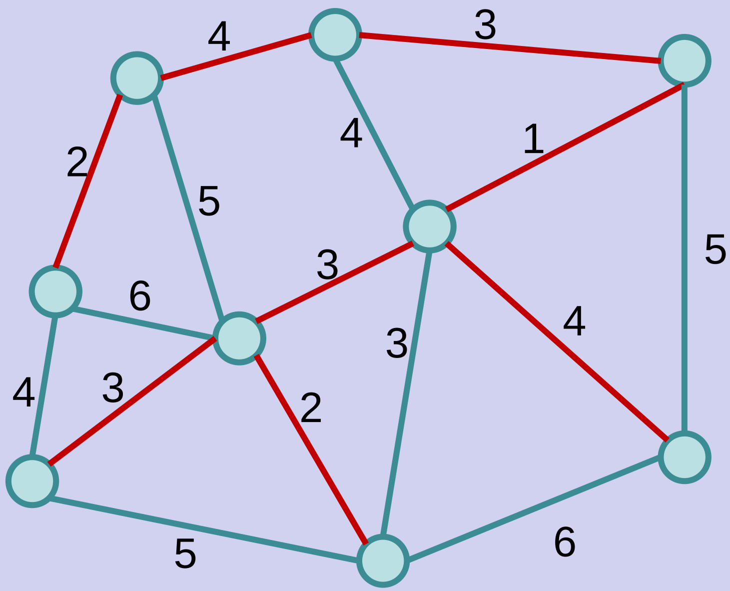
Partial + Full $\leq$ H + N / K $\leq$ 2 Opt

MST Bound for TSP

Undirected graph satisfying triangle inequality

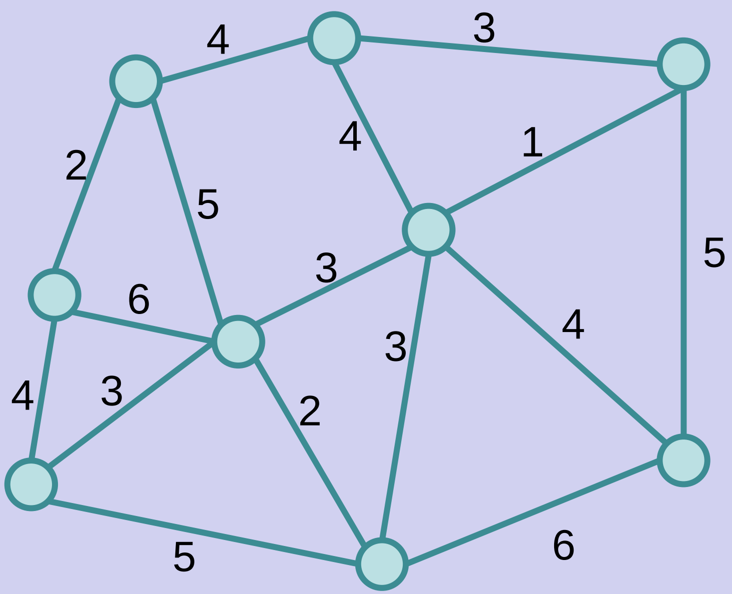
MST Cost ≤ TSP Cost ≤ 2 MST Cost





Christofides TSP Algorithm

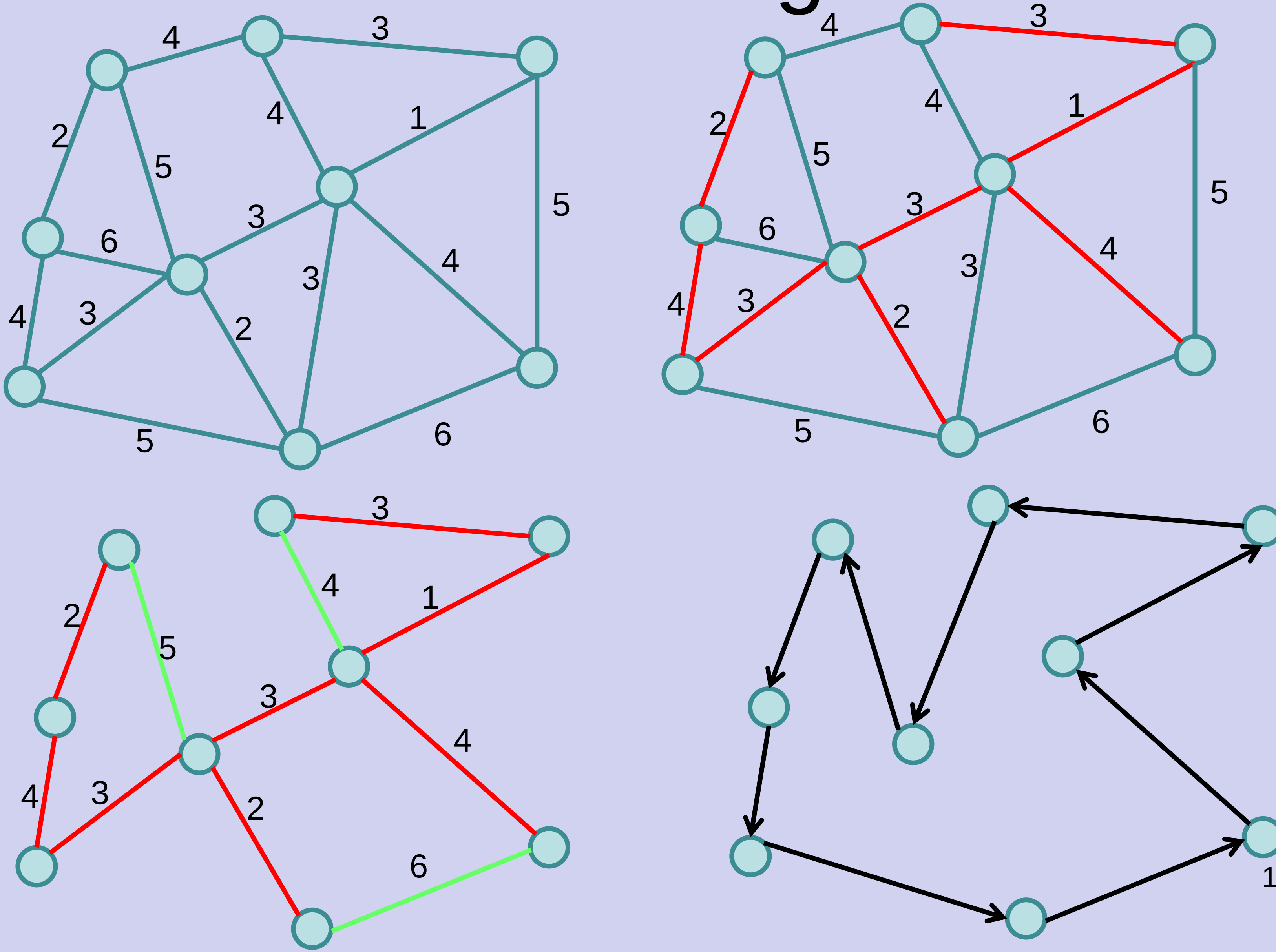
- Undirected graph satisfying triangle inequality



1. Find MST
2. Add additional edges so that all vertices have even degree
3. Build Eulerian Tour

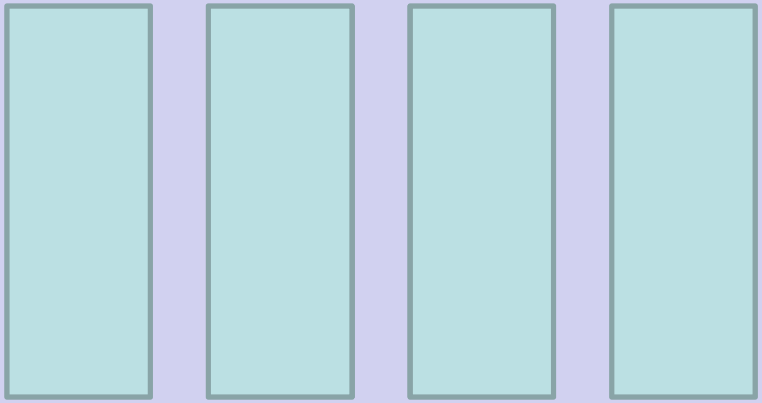
3/2 Approximation

Christofides Algorithm



Bin Packing

- Given N items with weight w_i, pack the items into as few unit capacity bins as possible
- Example: .3, .3, .3, .3, .4, .4



First Fit Packing

- First Fit
  - Theorem: FF(I) is at most 17/10 Opt(I) + 2
- First Fit Decreasing
  - Theorem: FFD(I) is at most 11/9 Opt (I) + 4

Knapsack

- Items $\{I_1, I_2, \ldots I_n\}$
  - Weights $\{w_1, w_2, \ldots, w_n\}$, Values $\{v_1, v_2, \ldots, v_n\}$
- Find set S of indices to maximize:
  - $\Sigma_{i \varepsilon S} v_i$ such that $\Sigma_{i \varepsilon S} w_i \leq K$
- Dynamic Programming solution:
  - Find the smallest set of a given value
  - Runtime O(nV) where V is the sum of the values
- Goal – for any $\varepsilon > 0$, we want a polynomial time algorithm that finds a solution of at least $(1-\varepsilon)$ Opt

PTAS (Polynomial time approximation scheme)

- Idea for approximation algorithm*
- Scale values so that $\frac{1}{2} \leq Opt \leq 1$
- Let $\varepsilon = 2^{-k}$
- Round the values down to multiples of ε^2
- Solve the DP using ε^2 values
- Runtime $O(n\varepsilon^2)$, Approximation $(1-\varepsilon)$

*Some details omitted in dealing with very small items.

Branch and Bound

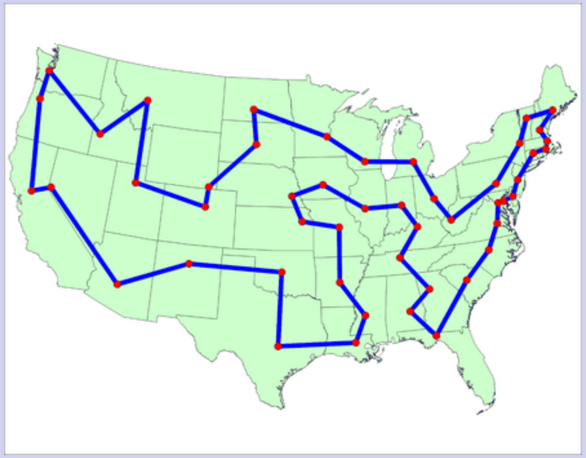
- Brute force search – tree of all possible solutions
- Branch and bound – compute a lower bound on all possible extensions
  - Prune sub-trees that cannot be better than optimal

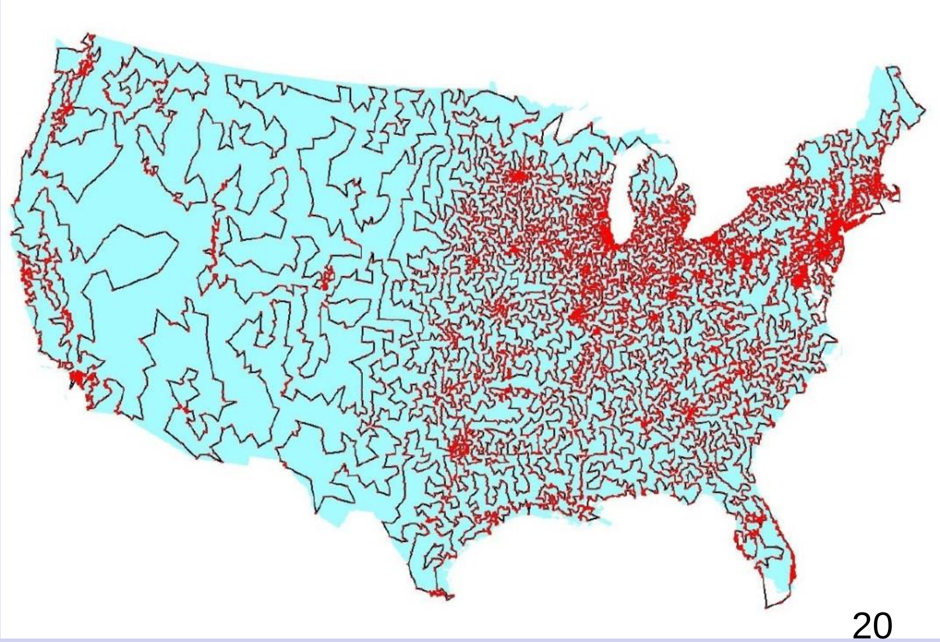
Branch and Bound for SAT

- Solving SAT by setting one variable at a time
- Setting a literal to 1 removes the clause
- Setting a literal to 0 removes the literal
  - Removing the last literal kills the subtree
- Heuristics for variable ordering
- Very important algorithms in practice, especially for software verification

Branch and Bound for TSP

- Enumerate all possible paths
- Lower bound, Current path cost plus MST of remaining points
- Euclidean TSP
  - Points on the plane with Euclidean Distance
  - Sample data set: State Capitals





Local Optimization

- Improve an optimization problem by local improvement
  - Neighborhood structure on solutions
  - Travelling Salesman 2-Opt (or K-Opt)
  - Independent Set Local Replacement

Enhancements to Local Search

- Randomized Local Search
  - Start from lots of places
- Metropolis Algorithm
  - Choose random neighbor
    - Move if cheaper
    - If worse, move with some probability
- Simulated Annealing
  - Like Metropolis, but adjust probabilities to simulate cooling