15.053 Thursday, May 9

- Heuristic Search: methods for solving difficult optimization problems

Handouts: Lecture Notes

See the introduction to the paper on Very Large Scale Neighborhood Search. (It's on the web site.)

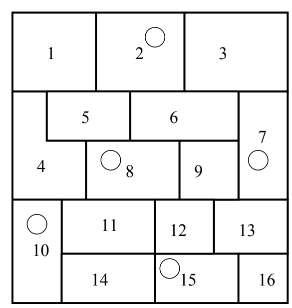
Two types of Complexity.

- **1. Problems with complex and conflicting objectives subject to numerous restrictions.**
  - **most problems in practice**
- **2. Problems that may be easily understood but for which there are so many possible solutions, one cannot locate the best one.**
  - **games (chess, go)**
  - **IPs such as the traveling salesman problem.**

Example: Fire company location.

- **Consider locating fire companies in different districts.**
- **Objective: use as few fire companies as possible so that each district either has a fire company in it, or one that is adjacent.**

Example for the Fire Station Problem



Reason for heuristics.

- **Heuristics are usually much faster than optimization, such as branch and bound**
- **Heuristics, if well developed, can obtain excellent solutions for many problems in practice**
- **Some special cases of heuristics**
  - **Construction methods**
  - **Improvement methods**

A construction heuristic for the TSP

begin
choose an initial city for the tour;
while there are any unvisited cities,
then the next city on the tour is the nearest unvisited city;
end

Construction heuristics: carries out a structured sequence of iterations that terminates with a feasible solution. It may be thought of as building a tour, but the intermediate steps are not always paths.

Illustration for TSP

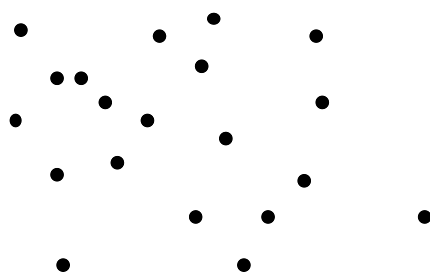
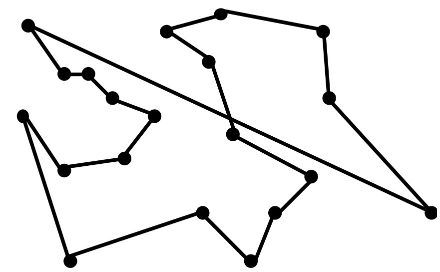


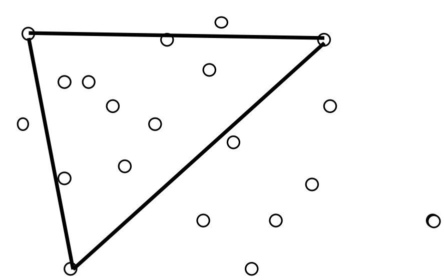
Illustration for TSP



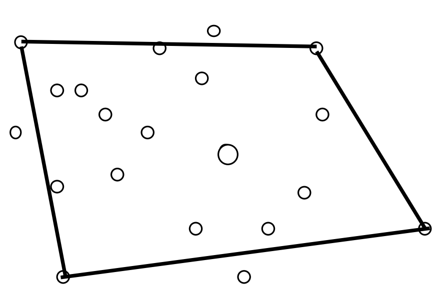
A more effective but slower construction heuristic

- The previous heuristic always added the next city at the end of the current path.
- Idea: add the next heuristic anywhere in the current path
- Better idea: keep a cycle at each iteration and insert the next city optimally into the cycle
- This is an insertion heuristic

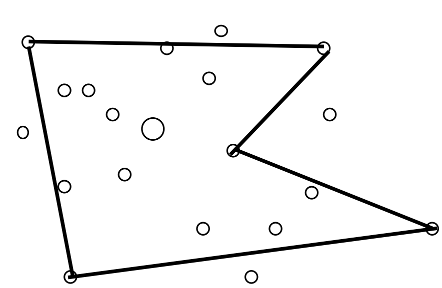
Start with a tour for 3 cities



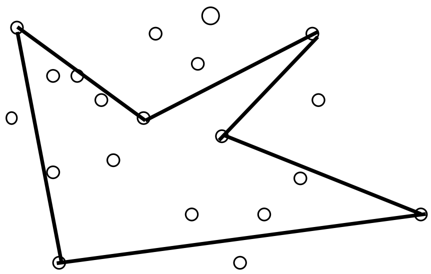
Insert the 4th city



Insert the 5th city

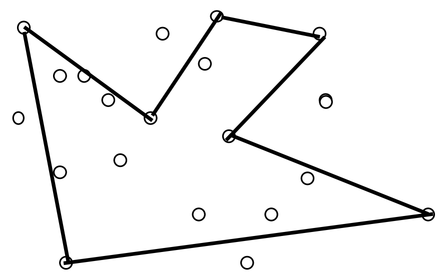


Insert the 6th city



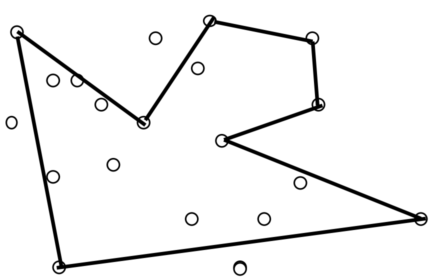
13

Insert the 7th city



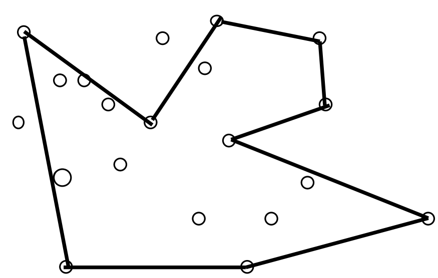
14

Insert the 8th city



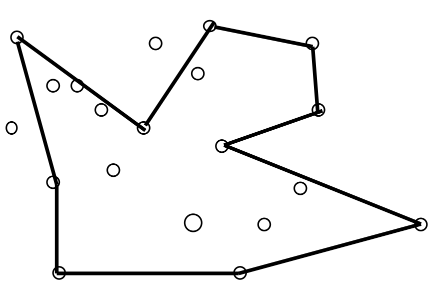
15

Insert the 9th city



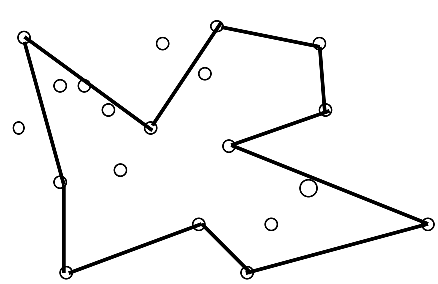
16

Insert the 10th city



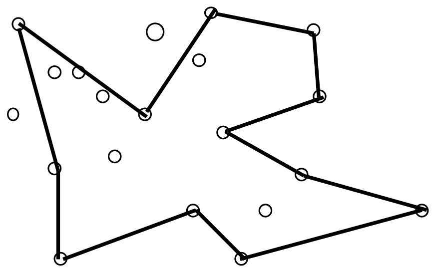
17

Insert the 11th city

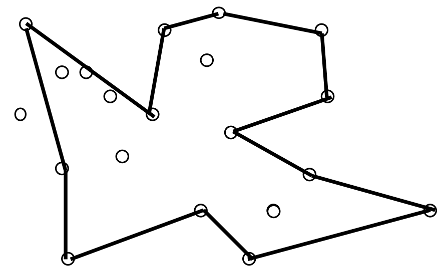


18

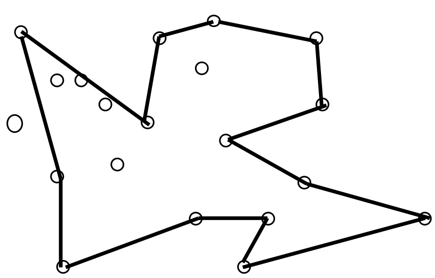
Insert the 12th city



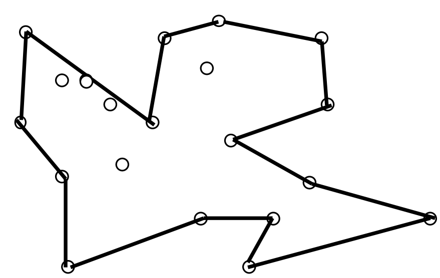
Insert the 13th city



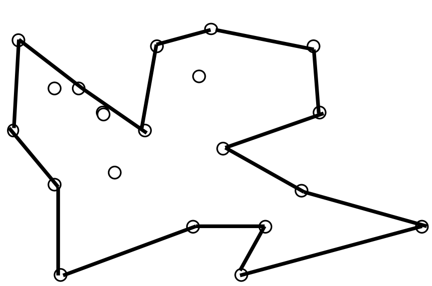
Insert the 14th city



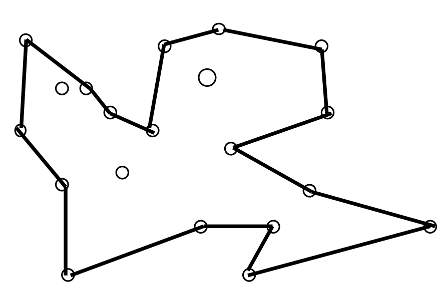
Insert the 15th city



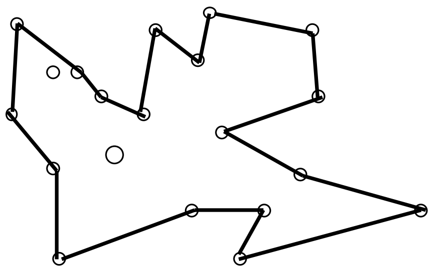
Insert the 16th city



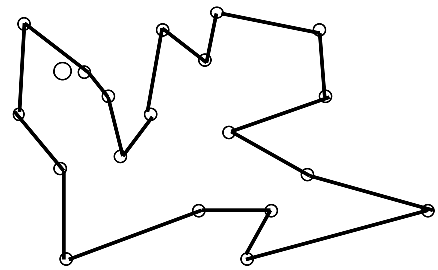
Insert the 17th city



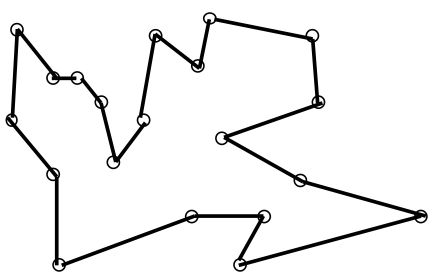
Insert the 18th city



Insert the 19th city



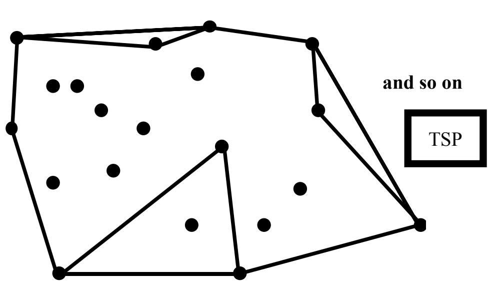
Insert the final city



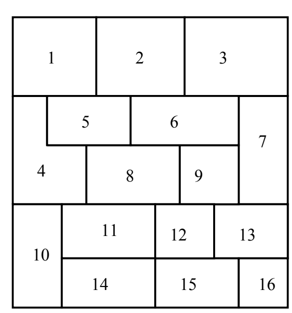
Comments on insertion heuristic

- Much slower than nearest neighbor
- Much more effective than nearest neighbor
- Choice of what city to insert makes a difference
  - inserting the city farthest from the current tour is most effective

Convex Hull + Insertion Heuristics



How can we do a construction heuristic for the for the fire station problem?



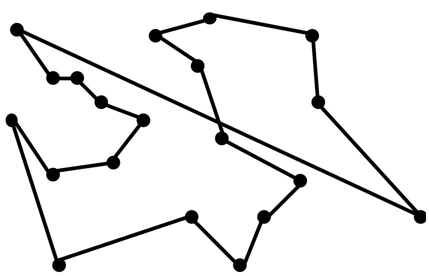
Some comments on heuristics

- It is easy to write satisfactory construction heuristics
- It is difficult to write good ones
- Sometimes simple is better

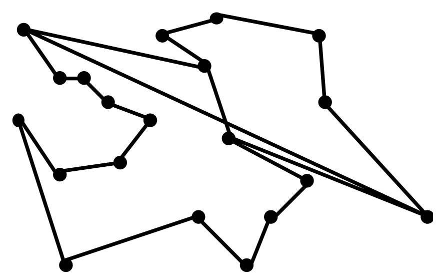
Improvement Methods

- These techniques start with a solution, and seek out simple methods for improving the solution.
- Example: Let T be a tour.
- Seek an improved tour T' so that |T - T'| = 2.

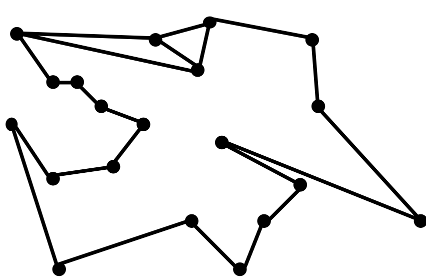
Illustration of 2-opt heuristic



Take two edges out. Add 2 edges in.



Take two edges out. Add 2 edges in.



Local Improvement Heuristic

- For any tour T, we say that T' is a ***2-neighbor*** of T if T' can be obtained from T by adding two edges and deleting two edges.
- We say that T is ***2-optimal*** if the length of T is less than or equal to the length of each of its 2-neighbors.

2-opt algorithm

begin with a feasible tour T

while T is not 2-optimal replace T by a 2-neighbor of T that has a lesser length.

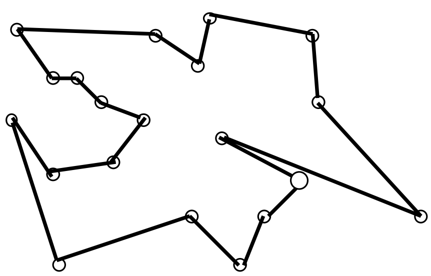
Comments on 2-opt search

- 2-opt generally produces good solutions, but it is not guaranteed to.
- It always eliminates the crossing edges
- It is typically within 7% of optimal.

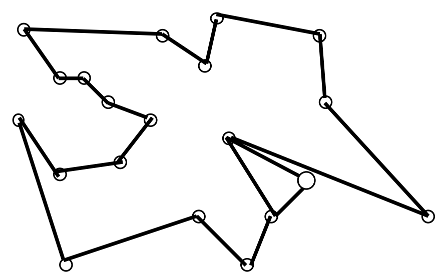
More on local search

- The basic principle: define a neighborhood of each possible solution.
- Given a solution x, replace x by a neighbor of x with lower cost, if one exists.
- The neighborhood often is specific to the type of problem at hand, and there are often many possible choices.
- Other possible neighborhoods for TSP
  - 3-opt
  - insertion

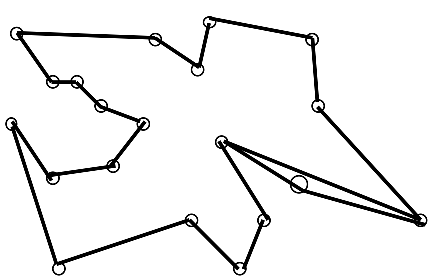
Insertion neighborhood: remove a node and then insert it elsewhere



Insertion neighborhood: remove a node and then insert it elsewhere



Insertion neighborhood: remove a node and then insert it elsewhere



Local Optimality

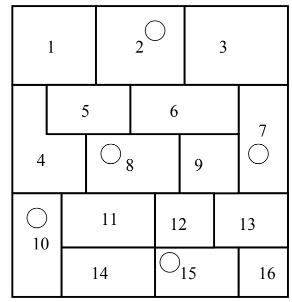
- A solution y is said to be ***locally optimum*** (with respect to a given neighborhood) if there is no neighbor of y whose objective value is better than that of y.
- Example. 2-Opt finds a locally optimum solution.

Improvement methods typically find locally optimum solutions.

- A solution y is said to be ***globally optimum*** if no other solution has a better objective value.
- Remark. Local optimality depends on what a neighborhood is, i.e., what modifications in the solution are permissible.
  - e.g. 2-interchanges
  - e.g., 3-interchanges

TSP

What is a neighborhood for the fire station problem?



Heuristics rarely come with performance guarantees

- They can be unpredictable.
  - 2-opt for the TSP is typically within a few per cent of optimum; but, it may be off by 100% or more.
  - A very stupid heuristic will occasionally outperform a far better heuristic (even a randomly selected tour could be optimal.)
  - One cannot predict how many iterations a local improvement heuristic will take.
  - To develop a good heuristic often requires "algorithm engineering"

Implementing Heuristics

- It helps to really appreciate algorithm design and implementation
- One can implement 2-interchange and 3-interchange for TSP in blindingly fast ways.
  - Problems with millions of "cities" have been solved, assuming that distances are Euclidean.

Randomization

- One of the most powerful ideas in heuristics and algorithms is randomization.
- In heuristics: this permits us to run essentially the same heuristic many times, and get many different answers. (Then one can choose the best.)

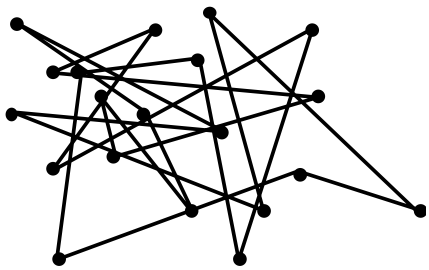
Insertion heuristic with randomization

Choose three cities randomly and obtain a tour T on the cities

For k = 4 to n, choose a city that is not on T and insert it optimally into T.

- Note: we can run this 1,000 times, and get many different answers. This increases the likelihood of getting a good solution.

A dumb-looking way to use randomization: choose edges randomly, one at a time.



Even dumb looking approaches can be of value

- Random tour followed by 2-opt.
  - Construct a tour by visiting cities in random order, and then run 2-opt. Repeat 1000 times.
- This works much better in practice than running 2-opt once. (In practice: starting from a random tour is slower than starting from a good tour, and so this technique is not used much.)

Simulated Annealing: a clever approach for using randomization

- Local improvement heuristics stop at a local optimally solution.
- Issue: is there a way of exploring a wider space. What if a locally optimal solution is a bad local optima.
- Simulated annealing is an approach for using randomization to occasionally make moves in the wrong direction.
  - based on a physical analogy
  - converges to the optimal solution under the right conditions

Simulated Annealing: a clever approach for using randomization

- Based on annealing, the cooling of some material to a "ground state," a state of minimum energy.
- Imagine taking a material that is very hot and cooling it slowly so that the material slowly hardens into the minimum energy state
- Fact: if one cools a material too quickly, the material will harden in some suboptimal configuration.

A quasi-translation of annealing to neighborhood search techniques.

1. T denotes a temperature
2. x denotes a current solution.
3. Find a neighbor y of x

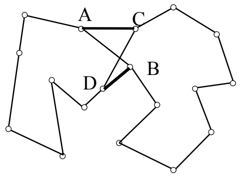
   If y is better than x, then let y be the new current solution.

   If y is worse than x by an amount Δ, then replace x by y with probability $e^{-\Delta/T}$.

Convergence to the optimum for simulated annealing

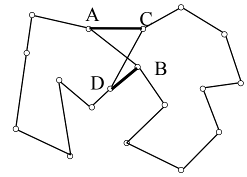
- Probability of a move in the wrong direction is $e^{-\Delta/T}$.
- As $T \to \infty$, Prob(wrong way) $\to 1$.
- As $T \to 0$, Prob(wrong way) $\to 0$.
- Simulated annealing gently lowers T from ∞ to 0.
- In theory, it converges to the optimal solution

Illustration of Simulated annealing



Simulated annealing will select a neighbor of T by randomly select two edges to leave the tour.

Illustration of Simulated annealing

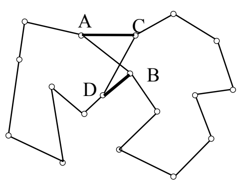


Suppose that the length of the neighbor is greater by Δ = 7

Probability of "accepting" the move is $e^{-7/T}$.

Simulated annealing will select a neighbor of T by randomly select two edges to leave the tour.

Illustration of Simulated annealing



Suppose that the length of the neighbor is greater by Δ = 7

Probability of "accepting" the move is $e^{-7/T}$.

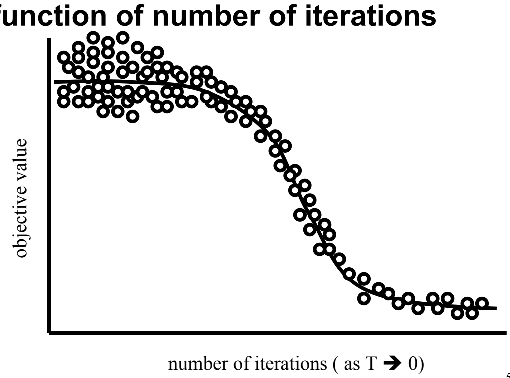
If T is close to 0, then the move will be rejected.

If T is very large, then the move will be accepted.

Simulated annealing in practice

- If one lowers the temperature slowly enough, the solution converges to the optimum with high probability (but one needs to lower the temperature excruciatingly slowly.)
- In practice, one lowers the temperature sort of slowly.
- For many problems, simulated annealing is excellent.

The average objective in SA as a function of number of iterations



Summary

- Construction Methods
- Improvement methods
- Randomization
- Simulation Annealing