Wednesday, March 30 – Day 26

Topic 11 - Hill Climbing

Next week: video lectures, will email

With Gradient Ascent as our inspiration, we want to think for ways to search for global optima in cases where the search space is:

(1) discrete
(2) continuous

Problem Setup:

* Search space S of possible candidates
* We may have some constraints
* Scoring function: score(x), $x \in S$
  (also called "fitness" or "quality")
* A way to generate either:
  - all the candidates "near" some given candidate; this is called "the neighborhood", nbhd(x).
  - a random candidate near a given candidate, a "tweak", tweak(x)

doesn't make as much sense in cns. problems

"near" is for you to define, it depends on the problem, and the algorithm you're using.

Two running examples:

(1) TSP (n cities)

* discrete — size of the search space: $(n-1)!$
* Score(T) = cost of the tour (sum of the weights) - we want to minimize
* nbhd(T)

Suppose $T = C_1 \to C_2 \to C_3 \to \cdots \to C_n \to C_1$

Define the neighborhood of T to be all ways of picking two cities and swapping them (excluding C_1)

nbhd($A \to B \to C \to D \to A$)

$= \{ A \to C \to B \to D \to A,$
$A \to B \to D \to C \to A,$
$A \to D \to C \to B \to A \}$

* tweak(T) = a random thing in nbhd(T)

Let's say we have n cities. How big is nbhd(T)?

$$\binom{n-1}{2} = \frac{1}{2}(n-1)(n-2) \qquad O(n^2)$$

(2) optimizing a continuous function in two variables $f(x,y)$

* continuous search space $\mathbb{R} \times \mathbb{R}$
  $[0,1] \times [0,1]$
* score of a candidate is the value of the function at that point
* nbhd$((x,y))$ = all points within some fixed distance δ of (x,y)
* tweak$((x,y))$ = pick something from nbhd$((x,y))$

MH #1: Random Search

```
best = random element of S
while True:          (quit whenever you want)
    x = random element of S
    if score(x) > score(best):
        best = x
```

Possible stopping conditions:

* best score does not improve for N iterations
* preset # of iterations
* you get impatient

This is not a good metaheuristic. It doesn't use any old information to guide future choices.

[2 demos]
- 01: TSP random
- 02: cns func 1- random

Gradient Ascent inspires the next one.

MH #2: Steepest Ascent Hill-Climbing (discrete only)

```
x = random element of S
while True:
    N = nbhd(x)
    S = element of N with the best score
    if score(s) > score(x):
        x = S
    else:
        quit
```

What does this do? Climbs right up the hill you start on.

Pros

* Finds a local optimum.

Cons

* Unlikely to find a global optimum unless your search space is very nice.
* very slow, especially if the neighborhoods are big, like TSP

Demos:

03 - TSP Steep. Asc. swap 2 50 cities

04 - TSP Steep. Asc. swap 2 300 cities

What's the slow part?

(1) generating the neighborhood

(2) scoring everything the neighborhood

$$d((x_1,y_1),(x_2,y_2)) = \sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

To score with 300 cities, we do 300 distance calculations

(two subtractions, two squarings, one addition, one square root)

This is slow when we have to do it

$\binom{299}{2} = 44,551$ times per move.