Mon, Apr 29, 2024

* HW 6 due on Friday
* Course Evals open

Topic 13 - Tabu Search

Pseudocode:

```
generation=0
taboo= dict()
taboo_time = 20   # depends on the problem
x = random elt. of search space
while True:
    generation += 1
    neighbors = nbhd(x)   # each neighbor is a
                          pair (s,m) where s is
                          the new sol (the neighbor)
                          and m is the "move"
                          that turned x->s
    new_x, move = the pair (s,m) in neighbors
                  with the highest score subject
                  to the constraint that either
```

```
            m is not a key in taboo
            OR taboo[m] ≤ generation
    x = new_x
    taboo[move] = generation + taboo_time
```

Advanced Topics

* "Aspiration Criteria" You can decide ahead of time to ignore the taboo list in some cases, for example if a taboo solution would be a new best record.

* Sometimes the neighborhoods are too big. You can try to generate a fixed # of random tweaks (100? 1000?) and pretend that's the whole neighborhood.

* Add extra exploration or exploitation depending on the performance of the algorithm.

- Keep track of how often components of solutions are being used in good or bad or any candidate.

Ex: How often a particular item is used in the 100 best or 100 worst knapsack solutions.

If you want more exploitation:
Force the items in the best sols to stay.
If you want more exploration:
Ban the good items.

Topic 14 - Neighborhoods in Continuous Space

So far we've used a simple tweak in our continuous problems.

$x = (x_1, x_2, \dots, x_d)$ d dimensions

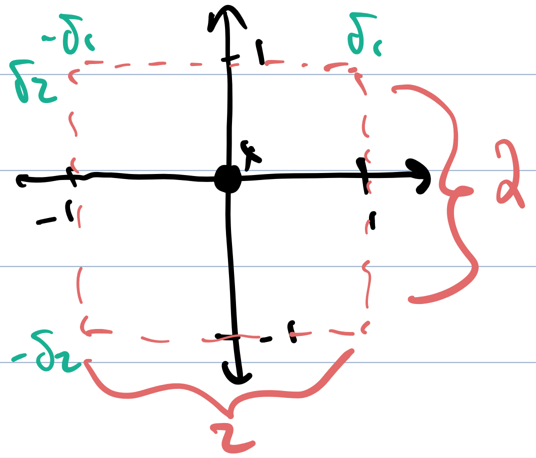
$s = \text{tweak}(x) = x + (r_1\delta_1, r_2\delta_2, \ldots\ldots, r_d\delta_d)$ where each r_i is a uniform random # in $\boxed{[-1,1]}$ and each δ_i is decided ahead of time.

Works okay when d is small (2 or 3)

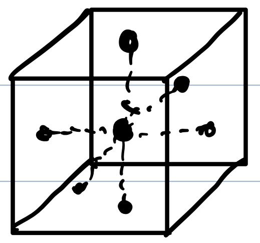
For now, assume $\delta_i = 1$ and $x = (0,0,0,\ldots,0)$ so $\text{tweak}(x) = (r_1, r_2, \ldots, r_d)$.

The new point tweak(x) is somewhere in the d-dimensional cube with side length 2 centered at the origin.

$d=2$



$d=3$



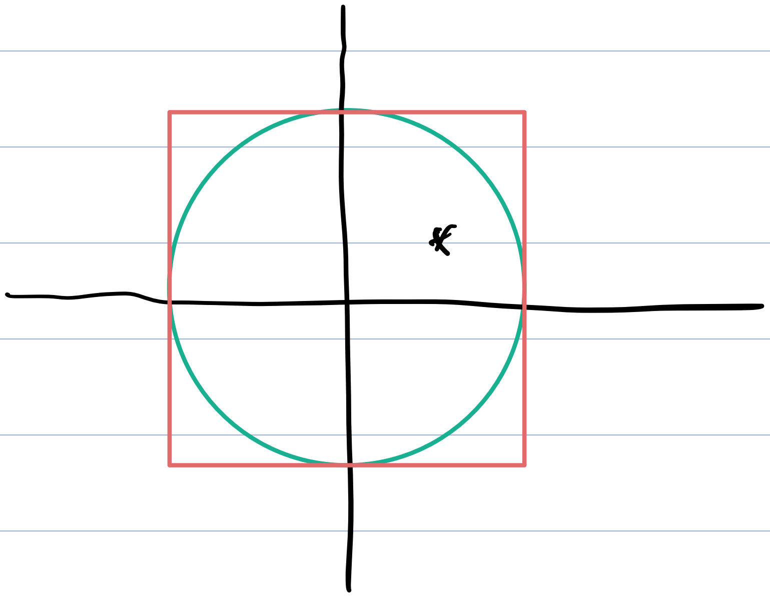
What is the farthest that s could be away from x?

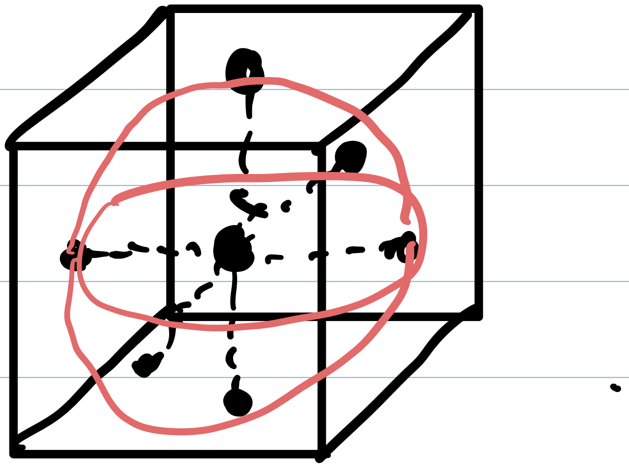
$d = 2 \rightsquigarrow \sqrt{2} \approx 1.41$

$d = 3 \quad \sqrt{3} \approx 1.7$

In d-dimensional space $\rightsquigarrow \sqrt{d}$

Picking points in a cube can lead to very far away tweaks. Instead pick from a sphere.





* How do you pick points uniformly from the unit circle?

Idea 1) Rejection Sampling

Pick points in the square

Check if each point is in the circle

- If so, keep it
- If not, throw away and try again

Works, but very slow in higher dimensions

Idea 2) Pick $x \in [-1, 1]$

Pick $y \in \left[-\sqrt{1-x^2}, +\sqrt{1-x^2}\right]$