CAS CS 440
Lec 5

Local Search (cont.)

1. Review
    a. Objective function converts everything to numbers to obtain the state that has the best value (goal) → we can draw curves (plot them)
        i. Each state will be drawn as coordinate
    b. The trouble is that we cannot plot them → since the world is more than 3D
        i. (state that the agent sees can have more than 3 elements)
        ii. Shape can be difficult to analyze
    c. Want our agent to move around the surface to try to find the max or min
2. Optimization with Local Optimizers
    a. Path on Objective surface = trajectory
    b. Local optimization only looks at "local" region to decide where to go next (only allowed to see something that is near the agent)
    c. Can get stuck
        i. Local optima (bad) → it is guaranteed to find local optima
        ii. How to get unstuck?
3. Hill Climbing
    a. Goal: move on the surface (to a goal state)
    b. Algorithm: when at state s:
        i. Generate all children of s, select child s' with "best" utility value
        ii. If s' is better than s, move to s' and repeat
        iii. Else give up and return s (do nothing)
    c. Greedy: can get stuck at local optima
    d. Fast & memory efficient:
        i. Generally easy to move improve state early on
        ii. Only keep current state in memory
            1. Best child search can be done with constant memory
    e. Problem: what happens if we reach a plateau?
        i. Hill Climbing will exit.
    f. Allow "sideways" move on plateaus (allow to move through plateaus → might do better if I move through them)
        i. Accomplished with thresholding (how to choose)
            1. If the best neighbor appears as good as the current state for x number of times, move x times to that direction before giving up (if the neighbor is as good as me for x number of times, stop there)
        ii. Accomplished probabilistically (flip a weighted coin)

1. If the neighbor is as good as the current state, move with x%
2. Probability of moving can depend on "steepness" of hill before plateau
    a. A version of momentum!

g. Random restarts: run multiple independent hill-climbers
    i. Different random initializations different trajectories
    ii. Return the "best" solution amongst trials

h. Reason you can't accomplish the best solution is because they are greedy algorithms that always tries to do better but there are times where you have to make worse decisions to achieve better results (short term loss can cause better result)

4. Simulated Annealing
    a. When should we go in "bad" directions?
        i. Short term loss can lead to long term gain
    b. Hill climbing always go in a "good" direction
    c. Simulated Annealing:
        i. Modify hill climbing
            1. Instead of picking the best move, pick a move at random (a candidate)
            2. If candidate is good, accept it (and repeat)
            3. If candidate is not good, accept with some probability, otherwise repeat
        ii. Scale prob of accepting a bad move as function of time
    d. It will either find the goal or get stuck – as time progresses simulated annealing starts to become vanilla hill climbing and starts to reject the bad choices with higher probability
5. Local Search In Continuous Spaces
    a. So far, algorithms proposed bad for continuous spaces
        i. (GA for model parameters works on continuous spaces)
    b. Problem: enumerating child states
        i. Potential infinite!
    c. Can still measure "good" and "bad" states
        i. Calculus not enumeration
    d. Can calculate the derivatives – it tells us direction
6. Derivatives
    a. If we have an objective which is continuous (or piecewise continuous)
        i. Can still optimize!
    b. Optima can be found using 1st (and 2nd derivatives)
        i. 1st derivative can tell us **where** optima is
        ii. 2nd derivative can tell us **what kind** of optima it is

7. Derivatives in Higher Dimensions
    a. Called a “gradient”
    b. Algorithm:
        i. For every variable in :
            1. Calculate the partial derivative (differentiate with respect to that single variable)
            2. Collect partial derivatives into a vector

$$\nabla f(x, y, z) = z^3 e^{(x+y)^2}$$
$$= \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}\right)^T$$
$$= \left(z^3 e^{(x+y)^2} 2(x+y), z^3 e^{(x+y)^2} 2(x+y),\ 3z^2 e^{(x+y)^2}\right)^T$$

    c.