Wednesday, April 28

Lecture #39/42

-> Homework Questions?

Topic 20 - Genetic and Evolutionary Algorithms

These will be population MHs inspired by evolution. Unlike PSO, Firefly, Cuckoo, the solution candidates don't need to represent points in space. These will work with any kind of space, continuous or discrete.

First group - "evolutionary strategies", we just do some tweeks to all of the things in our population, and keep the best ones.

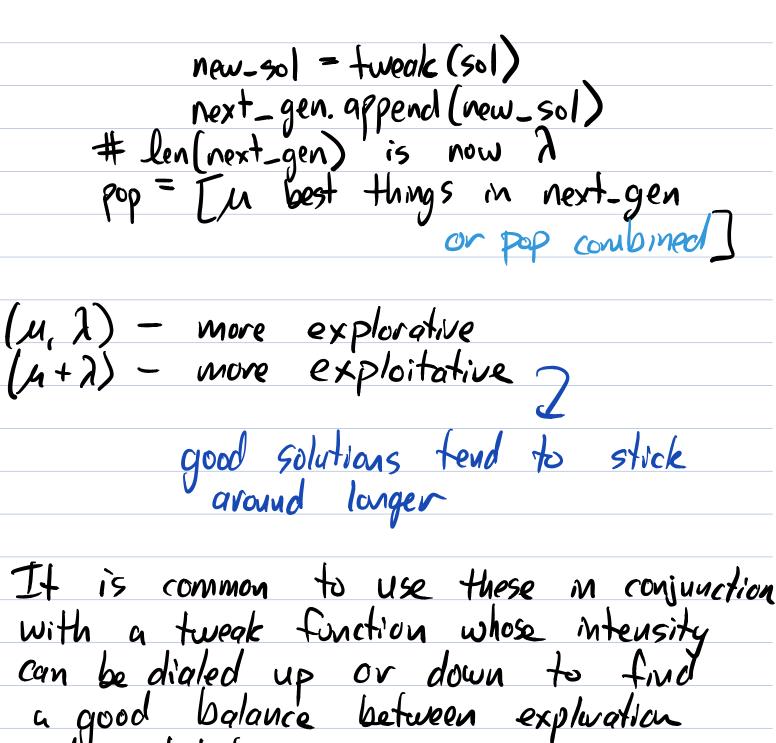
"(u, 2) Evolution Strategy"

* Starts with a population of a random

1)* Tweak each one of them 1/4

they are better worse. Now we have 2 solutions.
they are better worse. Now we
have 2 solutions.
* Keep the best u of those 7 and throw the rest away.
throw the rest away.
throw the rest away.
Pseudocode:
pop=[u vardom solutions]
while True:
best = best solution we've ever
Seen
next den = 57
for sol in son
next_gen = [] for sol in pop: repeat 2/n times: new_sol = tweak (sol)
now and = Lyant (sal)
new sol reduction
Mexit_gen. appendi (new_sol)
len(next_gen) is now 1
next_gen.append(new_sol) # len(next_gen) is now is now is now in next_gen] pop = [u best things in next_gen]
It's possible that nothing in next-gen
It's possible that nothing in next-gen is as good as the best thing in pop.

(u, \lambda) always throws away the u parents even if they were better than the \lambda children.
parents even if they were better
than the 7 children.
Variant: " (4+2) Evolution Strategy"
In this variant we stert with u
parents, generate 2 children, then we pick the best in solutions out of the u + 2 parents and children
we pick the best in solutions out
of the u + 7 parents and children
together.
Fy: (10+50)
together. Fx. (10+50) 10 parents -> 50 children -> pick 10 best out of 60
out of 60
Pseudocode:
pop=[u vardom solutions]
while True:
best = best solution we've ever
best = best solution we've ever seen
best = best solution we've ever seen
best = best solution we've ever seen
best = best solution we've ever



can be dialed up or down to find a good balance between exploitation.

"One Fifth Rule" "One nth Rule" * Aiming for about 1/5 of the children to be better than their porents. * If more, too much exploitation, dial

UP	explorati	tiw no	4 biggs	er tweaks.	
* It, 1	ess, too	much	explorati	ion, dial	
dow	n explor	ration w	th' sm	er tweaks. ion, dial maller tweaks	ς,

Examples: continuous functions

Gaussian walk - bigger or smaller

Standard deviation

TSP - k-opt with smaller or

larger k

Genetic Algorithms

Tied for the most famous MH w/ Simulated Annealing.

Adds one critical idea to the Evol.

Strots we just sow: crossover.

* Single parents can create children with a tweate (termnology: "mutation")

* Two parents can combine to produce one or more offspring that some qualities from each parent.

Big Idea: Stort with a population of a solutions. To form the next generation: we'll pick two solutions to be parents, then cross them over to form children. Each child is given some probability of mutating.

Do this a bunch of times with the a parents. Now you have a bunch of children. Pick the best ones, and they become the next generation.

Pseudocode:

pop = Lu random solutions]

while True:

best = best thing you've ever seen

next_gen = []

while len(next_gen) <):

select two povents Pi, Pz in pop

cross them over to get some children

allow each child to mutate with

some probability

add to next-gen

pop = [best in	4	next_gen]