Topic 17 - GRASP
Monday, May 2

Announcements:
- → HW 5 due the last day of class Mon, May 9 11:59pm
- → Final will be takehome, due Mon, May 16, 11:59pm

Topic 17 - Greedy Randomized Adaptive Search Procedures (GRASP)

Very fancy name for a very simple metaheuristic.

Idea: (1) Build a greedy solution, but not being as fully greedy as possible, so that you have choices.
(2) Starting at that greedy solution, perform H-C (single tweak or steepest ascent)
(3) Get rid of it, go back to (1).

Greedyish: At each step, compile some of the best next components, and randomly pick one of them to add.

How? Uniformly. Weighted (good = better or worse)

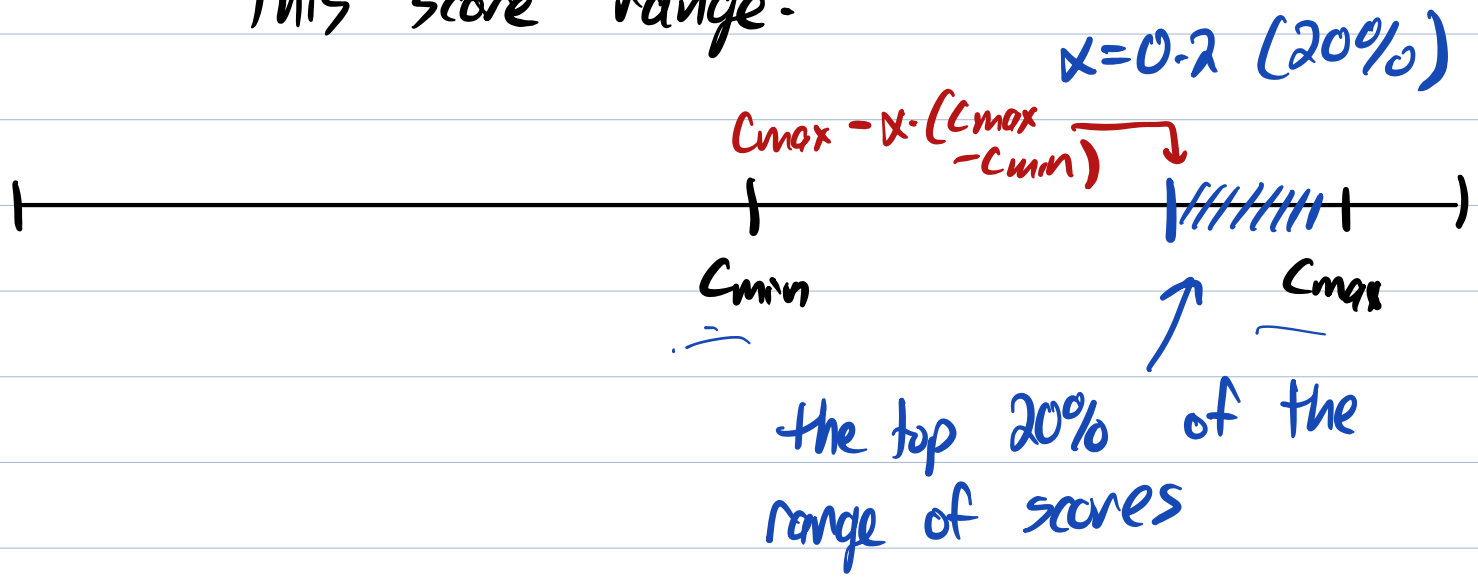
Called the Restricted Candidate List (RCL)

Two options for picking the RCL:

Option 1: Pick a percentage p and just put the top p% of options into the RCL. Good values of p depend on the problem, how many components you have. Usually 10% - 30% is good.

Option 2: Consider all possible next comp. and the value they would add. Let c_{min} and c_{max} to be the smallest and largest possible score. Form the RCL out of

all possibilities in top α% of this score range.



This corresponds to all components whose score is $\geq c_{max} - \alpha \cdot (c_{max} - c_{min})$

Option 2 is usually better. What should α be?

(A) A fixed value, maybe 10% - 30%?

(B) Each iteration, randomly pick some α from some range, maybe [0%, 30%].

(C) Make α an <u>adaptive parameter</u> (make it automatically adjust depending on the quality of solutions)

One way to do this:

Pick a finite set of α values:

$\{\alpha_1, \alpha_2, \ldots, \alpha_m\} = \{0.01, 0.02, \ldots, 0.30\}$.

Start with each being equally likely to be chosen,

$$p_i = \frac{1}{m} \quad \text{(in the example, } \frac{1}{30}\text{)}$$

Keep track of B = the best score ever and A_i = average score of all solutions found using α_i as the α value.

Every iteration, we recompute the p_i as follows:

Define $q_i = \frac{A_i}{B}$ (higher when avg. score is better, always ≤ 1)

Then define

$$p_i = \frac{q_i}{q_1 + q_2 + \cdots + q_m}.$$

This makes $p_1, p_2, \ldots, p_m$ that add up to 1, and are larger when A_i is larger.

Once the RCL is formed, how do we pick which component to use?

Standard answer: Pick uniformly from the options.

Other options: Bias toward better options
Bias toward the worse options

Topic 18 - Variations on Local Search

"Local Search" = MHs where we look nearby the current solution for new solutions.
H-C, SA, tabu search, etc.

Two variations to give you an idea of what's possible.

#1: Variable Neighborhood Search

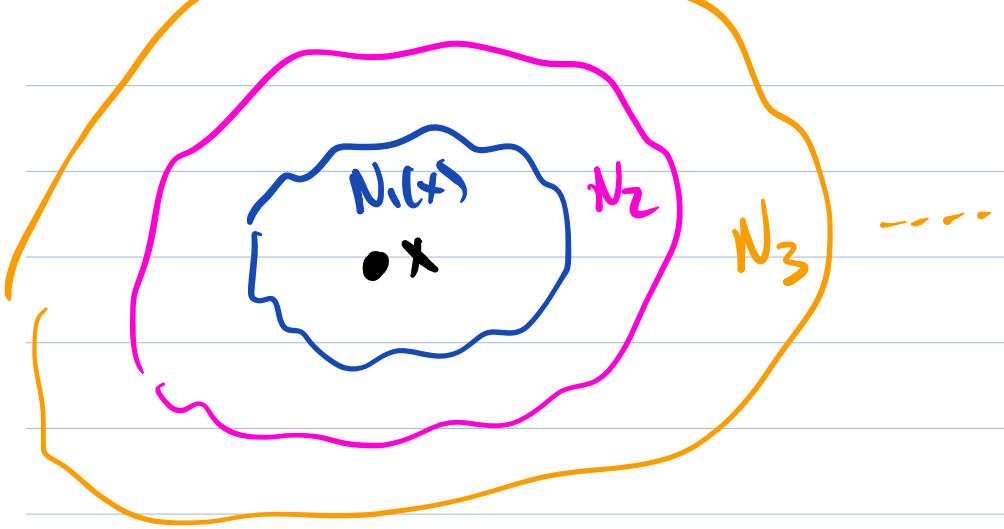
Idea: Define different kinds of tweaks (different neighborhoods)

$N_1(x), N_2(x), N_3(x), \ldots, N_d(x)$

→ more dramatic tweaks

Often the case:

$N_1(x) \subseteq N_2(x) \subseteq N_3(x) \subseteq \cdots \subseteq N_d(x)$



Procedure:

We have a current solution x

Pick $s \in N_1(x)$ randomly (s could be better or worse!)

Perform H-C on s to s'. (H-C = N_1 tweaks or something else)

If s' is better than x:

← Set x=s' and start over

Else:

Pick $s \in N_2(x)$ randomly

Perform H-C on s to get s'

If s' is better than x:

← set x=s' and start over

Else:

Pick $s \in N_3(x)$ ------ and so on

- - -

So we need:

* A neighborhood structure
* A H-C tweak (could re-use N, or could use something else)