CS 461
Spring 2021
Program 2 – Simulated Annealing

For this program you'll be using simulated annealing to solve a simple optimization problem.

Wassamatta University has just put the finishing touches on their new residence hall, consisting of 50 4-person suites, and selected 200 students for moving in. The residence hall is single-sex, so any member of our 200-student pool can be a roommate with any other.

Students have been provided with a questionnaire--whether they're early risers or like to sleep in, do they like having a lot of social activity or do they need their quiet time, how important is it that their space be neat & tidy all the time, etc. Each student has been rated as to their compatibility as a roommate with each other student. This relationship is symmetric; given 2 students A and B, Compatibility(A, B) = Compatibility(B, A). Note that this is pairwise, and we're putting 4 people in a room.

The challenge is how to assign people to rooms such that compatibility is as good as possible. Initially start with arbitrary assignment—the first 4 students in the first room, next 4 students in the next room, etc. For our purposes, rooms are interchangeable; we're not going to worry about whether 4 friends in 1 room and 4 friends in another room are on the same corridor or floor.

The fitness of a roommate assignment is based on the total of the pairwise compatibility scores; there are 6 such scores (AB, AC, AD, BC, BD, CD). Thus the total fitness score for a room assignment has a range of 0-600. This total should be as low as possible—lower scores indicate better compatibility. Choose an initial value for the 'temperature' parameter *T* such that most changes, including those that result in poorer fitness, are accepted; if in doubt, use a higher value for *T.* Use proportional reduction for *T* (that is, T = 0.9T or 0.95 T, etc), every 2000 accepted changes or 20,000 attempted changes. Repeat until you have an iteration with 20,000 attempts and no successful changes identified.

Data format: Assume the students are numbered 0-199; each row is one student's ratings (students have a rating of 0 with themselves). So the rating at position [a][b] is the compatibility between student a and student b.

Changes to apply: Choose randomly (uniformly) between two possible variations:

- Select 2 rooms at random, and 1 student at random from each room; exchange them.
- Select 2 rooms at random; swap the first 2 students in one room with the last 2 students in the other.

In both cases, your metric is changes in the total compatibility across both rooms.

Output: Produce a text file as output, reporting the best, worst, and average room compatibility score of your match, along with the parameters of your search (cooling schedule, initial value for T), followed by the room assignments. For convenience in reading the results, you may find it helpful to sort each room by student number; sorting rooms by lowest-numbered student assigned to that room would make it even easier to compare results.

Submit your source code and output file.

You may write your project in C, C++, C#, Java, or Python.