Ning

23 September 2019

Metaheuristics for Optimization

Class Summary

[OPENING]: This class is about how to use strategies for optimization when there is no analytical solution to the optimization problem. Will be pretty math based presumably. Class will be really important since optimization is part of the core of Machine Learning.

[INFO]: Bastien Chopard is the professor, and he is also the director of the Computer Science program. He can be reached at [bastien.chopard@unige.ch](mailto:bastien.chopard@unige.ch)

[GRADING]: The final exam is an oral exam, worth two-thirds of the final grade. The TP portion of the grade is worth the remaining one-third. It was not disclosed how many TPs we will have, but it is presumably comparable to the other classes.

[SUMMARY]: We began class by looking at a snippet of a Google interview question that posits, “given four people in their respective locations as denoted by these points, what is the minimum total distance walked?” Given the question, the minimization can probably be represented by something like, , but taking a step back, we looked at a 1D representation: x1 ------------------x3-----------x2. In this example, the best point is x3, because no matter how much x1 and x2 shift, one’s gain is another’s loss and vice versa, so the reasonable choice is that x3 doesn’t move. This point (that x3 happens to be on) is called the “median.” Professor Chopard then announces that to begin to answer the problem presented before, one should suggest that the optimal is the median between all the points given, then proceed to find a solution to obtaining the median value. That’s no formula(analytical) way to compute the median, so we’ll have to use an iterative computation.

[SUMMARY]: A *Search Space* is denoted as a set *S*, where it is the set that contains all the candidates that can optimize F. The *function* F takes the search space and outputs a real number. Formally, . Given a possible solution, xS, x is called the *configuration*. In the situation where (given that it is a minimization, reverse the inequality for maximization) then y is the optimal value. Note that there can be multiple *optimal solutions* , but only one *optimal value* (). A constraint is a requirement that an optimization has to meet. (ie: ). In this example, an easy way to meet the constraint is to simply replace your search space with . An alternative to replacing your search space is that you can penalize the fitness function accordingly. S can be discrete or continuous. In most of the considered problem, the search space is so large that exhausting all of the search space is too costly and unfeasible. When S is multidimensional, we call the number of dimensions the *problem size* or *degrees of freedom*. There are a couple of problems when we look at our search space. First, is that S grows exponentially with the problem size. Second, is that F is not always convex, and has local optimas, which perturb the optimization algorithm. In our case, *hard-optimization* problems require *metaheuristics* (cases where analytical computations don’t exist).

[NOTE]: There is a mention to a “Lagrange Multiplier” (which upon cursory look up, is a method to find the local optimas of a function subject to equality constraints. The idea is to convert the constrained problem into a problem that the derivative test still applies.)

[NOTE]: There is a mention to a type of problem called “Linear Programming” to which the solution algorithm called “Simplex” was mentioned. (Upon cursory look up, “Linear Programming” is simply a set of problems whose requirements can be modeled by linear relationships. Simplex here is about moving along the vertices of the linear relationships that creates a “polyptoe” which is a system of linear inequalities. Moving along the vertices until the optimal solution vertex is found.)

[SUMMARY]: I’ve attached some of the type of problems we looked at in class that I’ve taken by hand. They do not transfer well onto the computer so I have a scan of it. Check PDF: 2019-09-23 22.29.31