Assignment 2: Randomized Search

Submit Assignment

Due Nov 6 by 9:30am **Points** 100 **Submitting** a file upload
Available Oct 21 at 12am - Nov 6 at 9:45am 16 days

Assignment 2: Randomized Search

Released: 10/22/18
Due: 11/06/18, beginning of class

The assignment is worth 10% of your final grade.

Collaboration: You must complete and turn in this assignment individually. The “whiteboard policy" is in effect. (Please see the collaboration policy on the course website for details).

Overview

The purpose of this project is to explore random search. As always, it is important to realize that understanding an algorithm or technique requires more than reading about that algorithm or even implementing it. One should actually have experience seeing how it behaves under a variety of circumstances.

As such, you will be asked to implement or steal several randomized search algorithms. In addition, you will be asked to exercise your creativity in coming up with problems that exercise the strengths of each.

As always, you may program in any language that you wish insofar as you feel the need to program. As always, **it is also your responsibility** to make sure that we can actually recreate your narrative if necessary.

Read everything below carefully!

The Problems Given to You

You must implement three local random search algorithms. They are:

1. randomized hill climbing
2. simulated annealing
3. a genetic algorithm

You will then use the three algorithms to find good weights for a neural network. In particular, you will use them instead of backprop for the neural network you used in assignment #1. You may apply these algorithms to *one* of the datasets that you used in assignment #1 or a new dataset of your choosing. Notice that weights in a neural network are continuous and real-valued instead of discrete, so you might want to think a little bit about what it means to apply these sorts of algorithms in such a domain.

The Problems You Give Us

In addition to finding weights for a neural network, you must create (for sufficiently loose values of "create" including "steal", though it's fairly easy to come up with simple problems on your own in this case) two optimization problem domains. For the purpose of this assignment an "optimization problem" is just a **fitness function** one is trying to **maximize** (as opposed to a cost function one is trying to minimize). This doesn't make things easier or harder, but picking one over the other makes things easier for us to grade.

Please note that the problems you create may be over discrete-valued parameter spaces**. Using bit strings may make it more straightforward to apply genetic algorithms as they were presented in class.**

You will apply all three search techniques to these two optimization problems (in addition to the neural network problem). The first optimization problem should highlight advantages of your genetic algorithm, the second of simulated annealing. Be creative and thoughtful. It is not required that the problems be complicated or painful. They can be simple. For example, the 4-peaks and k-color problems are rather straightforward, but illustrate relative strengths rather neatly.

What to Turn In

You must submit a tar or zip file named **yourgtaccount**.{zip,tar,tar.gz} in t-square that contains a single folder or directory named **yourgtaccount** that in turn contains: -->

1. A file named **README.txt** that contains instructions for running your code
2. your code (this is critical: **if you don't turn in your code you will get a 0 on the assignment**)
3. a file named yourgtaccount-**analysis.pdf** that contains your writeup.
4. any supporting files you need (for example, your datasets).

The file yourgtaccount-**analysis**.pdf should contain:

- The results you obtained running the algorithms on the networks: why did you get the results you did? what sort of changes might you make to each of those algorithms to improve performance? Feel free to include any supporting graphs or tables. And by "feel free to", of course, I mean "do".
- A description of your optimization problems, and why you feel that they are interesting and exercise the strengths and weaknesses of each approach. Think hard about this.
- Analysis of your results. Why did you get the results you did? Compare and contrast the different algorithms. What sort of changes might you make to each of those algorithms to improve performance? How fast were they in terms of wall clock time? Iterations? Which algorithm performed best? How do you define best? Be creative and think of as many questions you can, and as many answers as you can. You know the drill.

Note: Analysis writeup is limited to 10 pages total.

Grading Criteria

At this point you are not surprised to read that you are being graded on your analysis more than anything else. I will refer you to this section from Assignment #1 for a more detailed explanation. I will also point out that implementing some of these algorithms is very easy (almost not worth stealing the code, but please feel free to do so anyway).

You should start now.