Evolutionary Computation

2016/2017

Notes for the PL 5

Ernesto J. F. Costa

March 7, 2017

Class PL 5

Evolutionary Algorithms (III)

5.1 General Goal

It is now clear that finding a solution for a problem with evolutionary algorithms involves many design issues. The most important ones are the choice of a proper representation and devising an appropriated set of variation operators (mutation and crossover) for that representation. During classes we have been focus on three types of representations: binary, reals and permutations. For each of them we defined mutation and crossover operators. To test the algorithms for each class we used benchmark problems: João Brandão's Numbers and 0/1 Knapsack for binary, optimization of continuous functions for reals, and the TSP for permutations. Once we have solved the two issues (representation and variation operators) we must concentrate on the problem of how we evaluate the quality of the solutions. And this is a problem dependent design problem. As we know the fitness of an individual is defined at the upper, phenotype, level, and thus there is the need of a so called representation function that translates from the lower, genotype, level, into the phenotype. Of course, if we use direct encoding that function reduces to the identity function.

During classes we presented many more representations, e.g., graphs, and proper operators for those representations. We also saw that the same problem may be encoded differently. Usually, as a general rule of thumb, the lower the representation the simpler the operators needed, but the the more complex the representation function. Some times there is a trade-off on this issue that is not easy to solve. Once we split the representation and the variation operators we may define a general evolutionary algorithm, that can be used for all problems of the same class. The only specific thing we have to

do is the implement the representation and the fitness functions. We know have simple code for each of the three representations, together with auxiliary functions to visualize the results and/or store the results into a file for posterior (statistical) analysis.

Our goal for today's Lab is twofold: (1) to use the general algorithms with new problems that were briefly discussed in class, and (2) to discuss and implement solutions for new problems. Let's do it!

5.2 Problems Revisited

Problema 5.1

For each of the problems mentioned below, choose the most appropriate representation and variation operators, define the representation and fitness functions, and use the proper general algorithm to test your solution. Do not forget to do several runs and visualize the results.

- N-Queens
- Satisfiability Problem (SAT)
- Neural Network Weights Learning

For the N-Queens problem the size of the board is a parameter of the problem.

For the satisfiability problem the number of variables is a parameter, and you have to define a specific formula expressed, as a conjunction of disjunctions of boolean variables, some negated, some not. When choosing the formulae be aware of the fact that it make sense to test your solution with formula for the two possible situations: satisfiable and unsatisfiable. Below an example of a formula in the conjunctive normal form: conjunction of disjunction of literals (boolean variables negated or not). You are free to choose whatever formula you want to test.

$$(x_2 \vee x_3 \vee \overline{x}_4) \wedge (x_1 \vee x_3 \vee x_4) \wedge (\overline{x}_1 \vee x_2 \vee x_4) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_2 \vee \overline{x}_3 \vee x_4) \wedge (\overline{x}_1 \vee \overline{x}_3 \vee \overline{x}_4) \wedge (x_1 \vee \overline{x}_2 \vee \overline{x}_4)$$

The choice of the fitness function may be tricky, for this is a yes/no problem. If you use a simple fitness function which map only these two situations

you will not be able to direct the evolution, because the induced fitness landscape transform your problem into the one of finding a needle in a haystack. A random search algorithm will have the same chance as your algorithm!

In the case of the Neural Network Weights Learning problem, you have to previously determine the topology of the network and what is the function the neural net is suppose to approximate. Lets us suppose that your function is the XOR, a boolean function that gives 1 if and only if both inputs are different. It can be implemented by a simple neural net as you can see in figure 5.1.

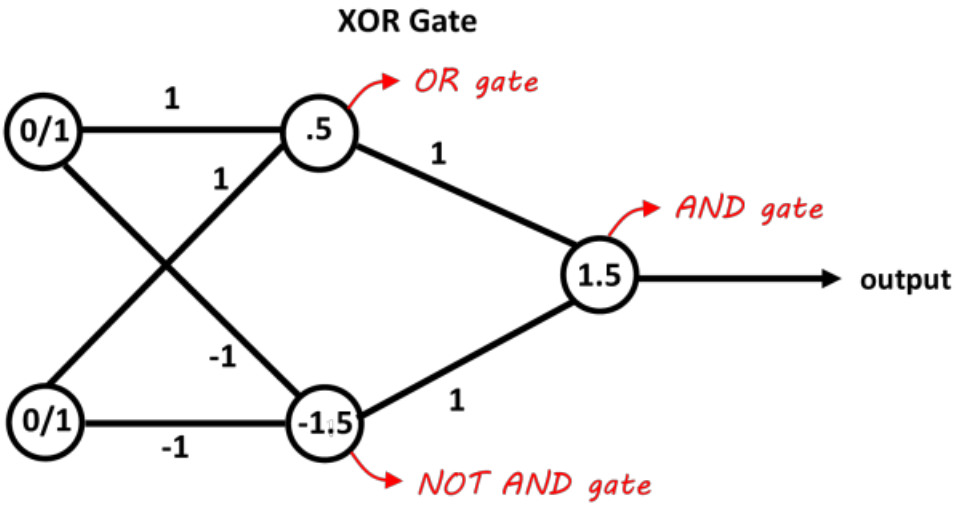
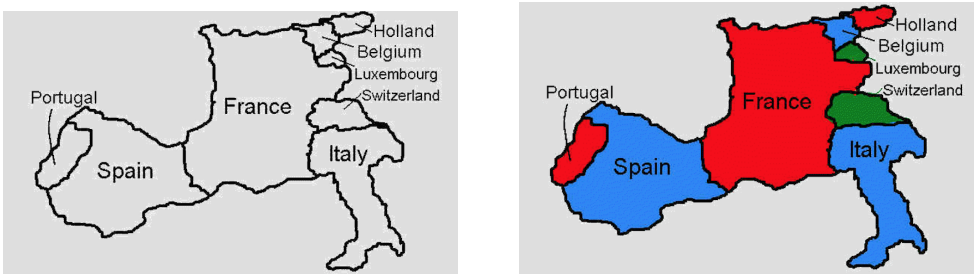


Figure 5.1: A neural network that implements a XOR

In the figure, the problem is already solved for the weights are defined. In your case these weights are unknown and the problem to solve is exactly to use an evolutionary algorithm to find them. Do not forget that the numbers inside the neurons are the values of the thresholds and they too have to be determined. Think carefully how your are going to measure the quality, the fitness, of each candidate solution.



(a) No color (b) Color

Figure 5.2: Map coloring problem

5.3 Map coloring

Problema 5.2

The map coloring problem can be defined as the problem of finding a proper color for each region (country) so that no two regions (countries) with a common boarder have the same color. See figure 5.2 for the example of the western Europe map.

Your task is to solve the problem with an evolutionary algorithm. If you have alternative representation try them all and compare the results (e.g., number of colors used, number of evaluations needed to find a solution, complexity of the variation operators, hardness of the representation function).