Solving the Traveling Salesman Problem with Genetic Algorithms

Daniel Kearney

March 20, 2018

Contents

0.1 Summary

0.2 History of the Genetic Algorithm

0.3 The Traveling Salesman Problem

0.4 Genetic Algorithm Function

The genetic algorithm works by mimicking the mechanism of chromosomes and genes in evolutionary biology. Individuals with various sets of genes compete with one another, reproducing into the next generation, with the whole population becoming fitter and fitter as the fittest solutions survive, and the less fit ones don't.

The genetic algorithm works by initially creating a random 'generation' of solutions to seed the iterative process. Each individual solution – called a 'chromosome' – has a random solution to the problem at hand.

Once the seed generation is created, the loop begins. The invariant is as follows. First, certain individuals are selected into the next generation, with the fittest solutions being favored. Pairs of these parents are 'crossed over', a process where parts of each solution are absorbed into each other. Finally, a bit of randomness is added by 'mutating' some of the solutions with random adjustments.

The loop has no natural termination point. The loop can terminate after a certain number of generations, or when a desired fitness is reached.

The following pseudocode summarizes how the genetic algorithm works. In the following section, the paper describes an implementation in Python.

```
current_generation = generate_random_solutions
for G generations:
    next_generation = current_generation.select
    next_generation.cross_over
    next_generation.mutate
    current_generation = next_generation
```

0.5 Genetic Algorithm Implementation in Python

The genetic algorithm can be implemented in numerous ways.

0.5.1 Creating the First Generation

0.5.2 Selection

0.5.3 Crossing Over

0.5.4 Mutation

0.6 Complexity Analysis

0.7 Results

0.7.1 Small Dataset

0.7.2 Large Dataset

0.7.3 Large Real-world Dataset