Using Genetic Programming to Implement Symbolic Regression

Caroline Thompson and **Rich Korzelius**
{cathompson,rikorzelius}@davidson.edu
Davidson College
Davidson, NC 28035
U.S.A.

Abstract

We use genetic programming to implement symbolic regression. Specifically, we use symbolic regression to determine a mystery function f. We do this by creating many potential functions and use genetic programming to find the best fit to the mystery function. We are still awaiting results to find the functions that best fit our data points.

1 Introduction

In this problem, we create a computer program to determine a mystery function by using symbolic regression. To do this, we use genetic programming to search for the best function that matches the mystery function. In other words, we create a population of possible functions and use genetic programming to determine which function best matches the mystery function. To determine which function best matches the mystery function, we assign each function a fitness value and use that to determine which function survives to the next generation. We use methods such as crossover and mutation to improve the fittest functions, as described by John Koza (**?**). We are still working on our results, which will be briefly summarized in this section. In the following sections, we discuss the background of this problem, our experiment, and our results.

2 Background

We later describe how we execute the reproduction, mutation, and crossover methods to produce generations, but we adapt a method called tournament selection to select individuals from Brad Miller and David Goldberg (**?**). In this method, we calculate the probability of individuals being selected using their fitness values. Then, we choose a sample of the population and select the individual with the best fitness value. We continue to use tournament selection to choose individuals for the various methods of determining best fit.

3 Experiments

We seek to find a mystery function using genetic programming.

First, we generate an initial population of possible functions. To do this, we randomly create functions expressed by the arithmetic operators +, -, x, /, positive integer powers of x, and integer constants. We choose an initial population size of 300 possible functions. We also choose a maximum depth of 10 for each function tree. We choose these parameters because they are large enough for us to get a large variety of possible functions, but small enough to run quickly and prevent the computer from running out of memory. We only use integers from -5 to 5 for the same reasons. To prevent the function calculating negative exponents, we take the absolute value of an integer power of x. When creating the initial population, each operator has an equal chance at being chosen for each equation. Thus, we have a wide variety of different functions.

Second, we give each function a fitness value to determine how closely its value matches the mystery function's value. For each individual, we evaluate the function value for a given x, and subtract this from the mystery function value of x. Then, we take the absolute value of this function and assign it as the individual's fitness value. We use the fitness value to determine the fittest individuals and choose individuals for crossover, mutation, and reproduction proportionately to their fitness values. Our program runs until an individual's fitness is as close to zero as possible, which indicated that we have found the mystery function.

Next, we select individuals for reproduction, mutation, and crossover. We decide on a 1 percent reproduction rate, 10 percent mutation rate, and 89 percent crossover rate. Rather than simply selecting the fittest individuals for these operations, we select individuals proportionately to their fitness values. This prevents possible diversity issues from arising because although fitter individuals are more likely to

be chosen, less-fitter individuals also have a chance at being chosen. As described previously, we use a tournament selection method to determine the best individuals for the operations. The first operation to be executed is reproduction. In this method, we select an individual and simply copy it into the next generation. The next operation is mutation. We select an individual and then randomly choose a node in the tree to mutate. Once we select a node, we delete the subtree after that point and reconstruct another tree using the same method we use to generate the initial population. Finally, we execute the crossover operation. In this method, we select two individuals and randomly choose either the right or left subtrees in each individual. Then, we swap the trees at these points in the two trees we selected.

After executing these operations a number of times, the trees get fitter and the fitness values get closer to zero, so we get closer to finding the mystery function with each generation.

4 Results

We are working on our results, which will be presented and analyzed in this section.

5 Conclusions

In this paper, we use genetic programming to implement symbolic regression. We perform symbolic regression on data points to determine a mystery function f. We use reproduction, mutation, and crossover to improve each generation's fitness and find the individual that best represents the mystery function. We are working on the results, which will be briefly summarized in this section. In further studies, we would expand our function span to include trigonometric functions, non-integer powers of x, and non-integer constants.