Rapid Evolution of Arithmetic Expression Snippets via Randomized Selection

Emilio Zangger
Virtual Computational Laboratory, Melbourne, Australia

Abstract

Randomly generated Python expressions composed of integers (1–10) and arithmetic operators (+, -, *, //, **) were iteratively scored using their evaluated numeric result. The top-performing snippets were selected, demonstrating a simple evolutionary selection framework applied to code generation.

Introduction

Automated code generation and evaluation have been explored in genetic programming. This experiment investigates a lightweight variant: generating small arithmetic expressions randomly and ranking them by output value, isolating the principle of stochastic code evolution in a minimal environment.

Methods

Expressions were randomly constructed using integers 1–10 and operators {+, -, *, //, **}. Each snippet was evaluated using Python's eval() function, and the output served as a performance score. Top-performing snippets were retained. Ten snippets were generated per run, with the top 3 selected.

Results

A typical run yielded top snippets such as: '9 ** 2 -> Score: 81', '7 * 8 -> Score: 56', and '10 + 10 -> Score: 20'. Even in a simple arithmetic context, stochastic generation followed by selection quickly identifies high-performing candidates.

Discussion

The experiment illustrates evolutionary search applied to code snippets: variation followed by selection based on fitness. While evolutionary computation is established, applying it in a concise arithmetic code context with direct evaluation as score represents a lightweight, interpretable demonstration.

Conclusion

Randomized generation and selection of arithmetic code snippets effectively identifies high-value expressions with minimal computation, highlighting principles of evolutionary programming, stochastic optimization, and fitness-based selection.