Chapter 10: Genetic Programming

Computational Intelligence: Second Edition

Contents

Compact Overview

- Developed by Koza in the late 1980s
- A specialization of GAs, where tree-based representations are used
- Originally developed to evolve computer programs
- As part of the fitness function, each program is executed to determine its outcomes

Implications of Tree Representations

- Each chromosome respresents a solution as a tree
- Implications:
  - Adaptive individuals
    - Size of individuals not fixed
    - Size depends on depth of tree and branching factor
  - Domain-specific grammar
    - A grammar needs to be defined that accurately reflects the problem to be solved
    - It should be possible to represent any possible solution using the defined grammar

The Grammar

- Terminal set specifies all the variables and constants
- Function set contains all the functions that can be applied to the elements of the terminal set
  - mathematical
  - arithmetic
  - Boolean
  - decision structures such as *if-then-else* and loops
- Semantic rules which characterize valid solutions

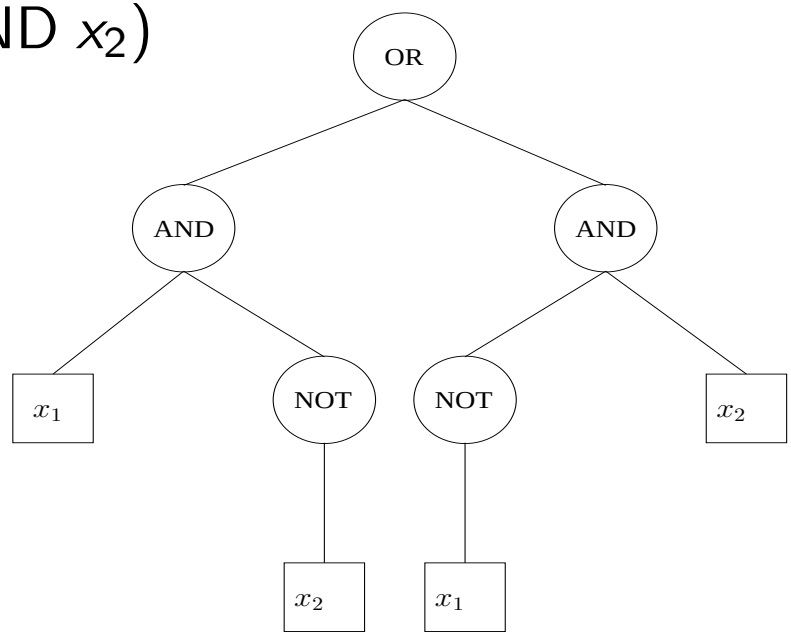
Example: Boolean Expressions

- Evolve Boolean expression

  (x_1 AND NOT x_2) OR (NOT x_1 AND x_2)

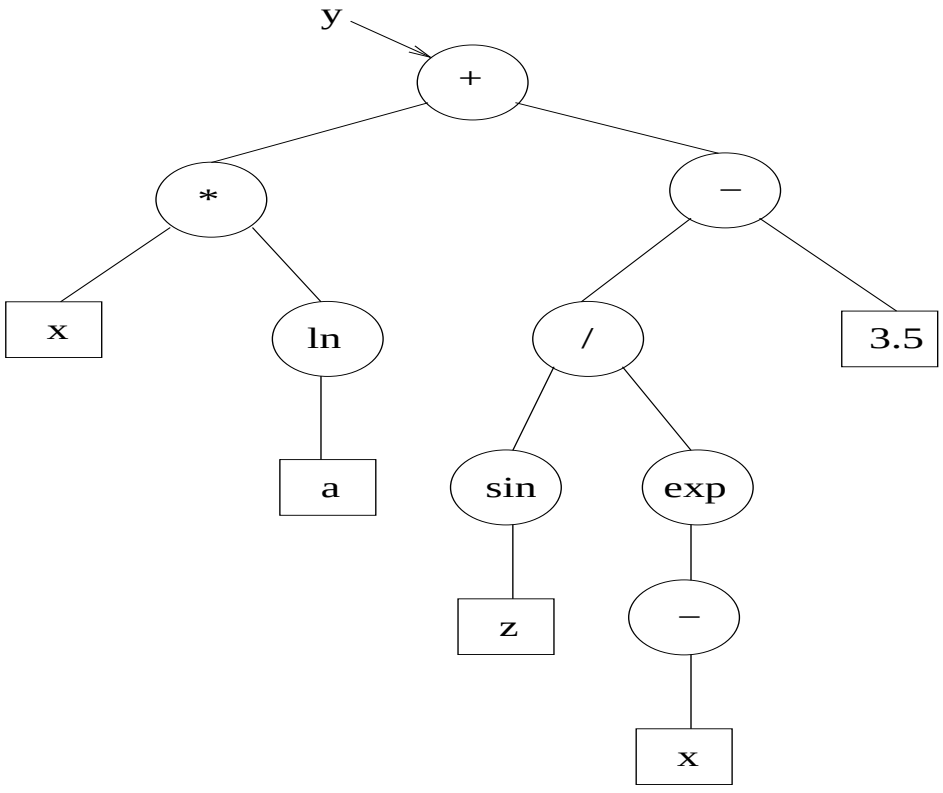
  given data set

x_1	x_2	**Target Output**
0	0	0
0	1	1
1	0	1
1	1	0



- Function set: $\{AND, OR, NOT\}$
- Terminal set: $\{x_1, x_2\}$ where $x_1, x_2 \in \{0, 1\}$

Example: Boolean Expressions



- Evolve mathematical expression

  $y = x*ln(a)+sin(z)/exp(-x)-3.4$

- Terminal set: $\{a, x, z, 3.4\}$ with $a, x, z \in \mathbb{R}$
- Function set: $\{-, +, *, /, \sin, \exp, \ln\}$

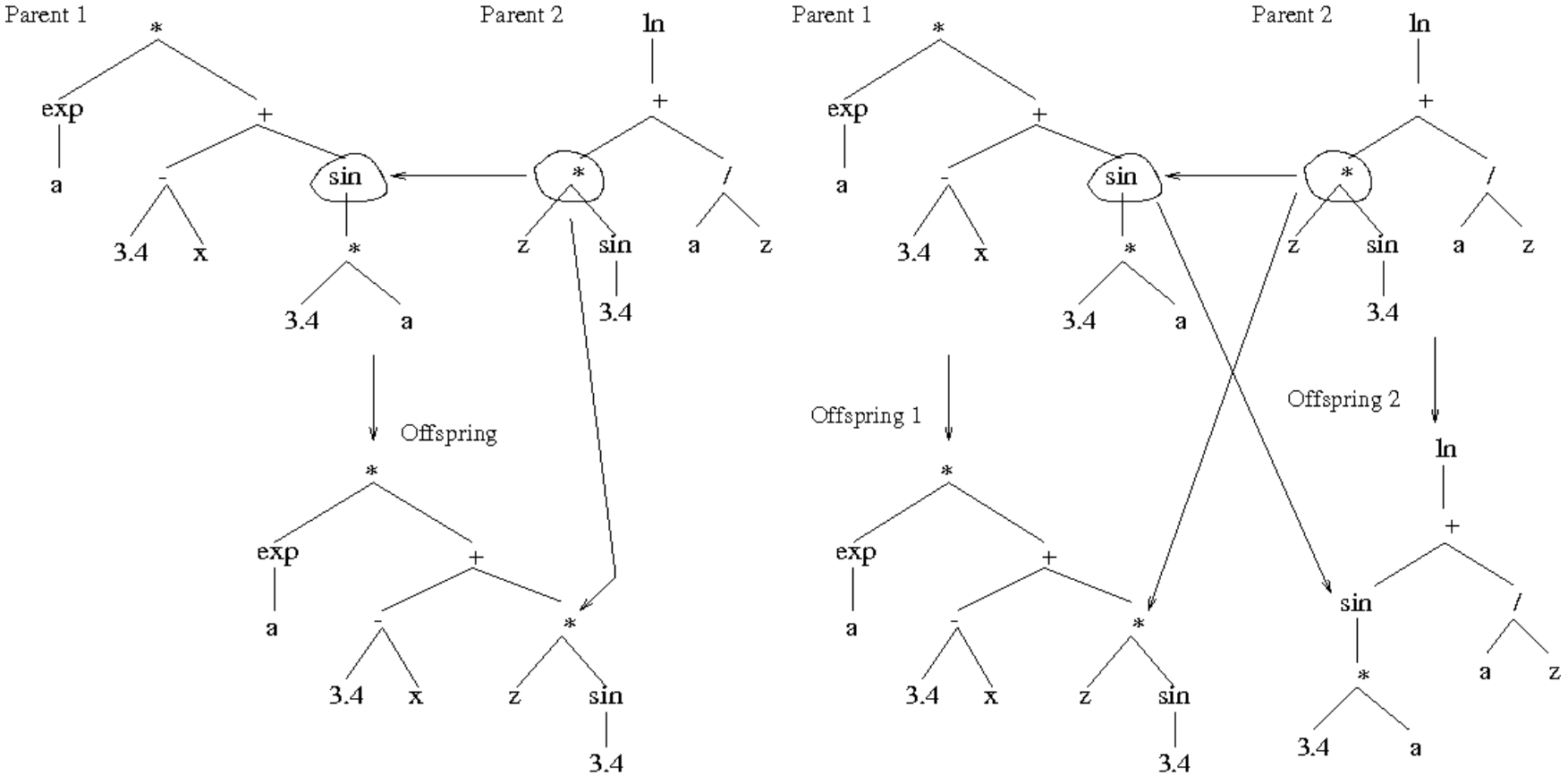
Initial Population

- Randomly generated within the restrictions of a maximum depth and semantics as expressed by the given grammar
- For each individual, a root is randomly selected from the set of function elements
- Non-terminal nodes are randomly selected from the function set
- Terminal nodes selected from the terminal set
- Start with small initial trees

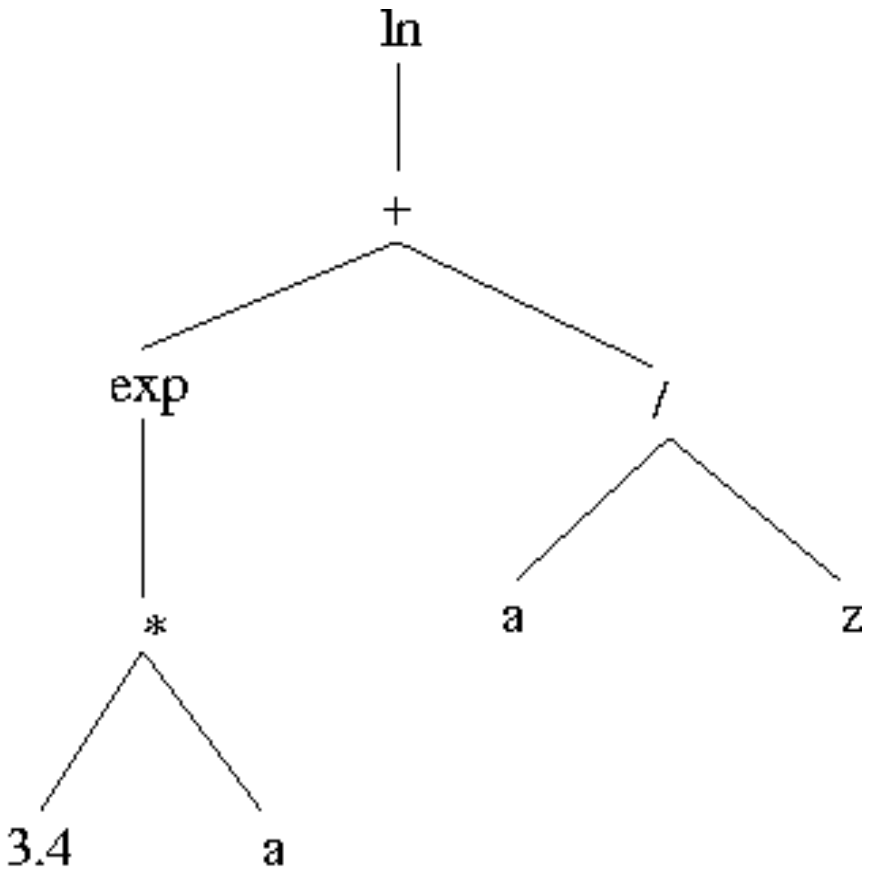
Fitness Function

- Fitness function is usually problem-dependent
- Fitness calculation requires the program to be evaluated against a number of test cases
- For the Boolean expression:
  - Fitness is the number of correctly predicted target outputs
- For the mathematical expression:
  - Fitness is the mean-squared error with respect to supplied target outputs
- Fitness function may include a penalty to penalize individuals with undesirable structural properties:
  - Penalize invalid solutions
  - Penalize tree size

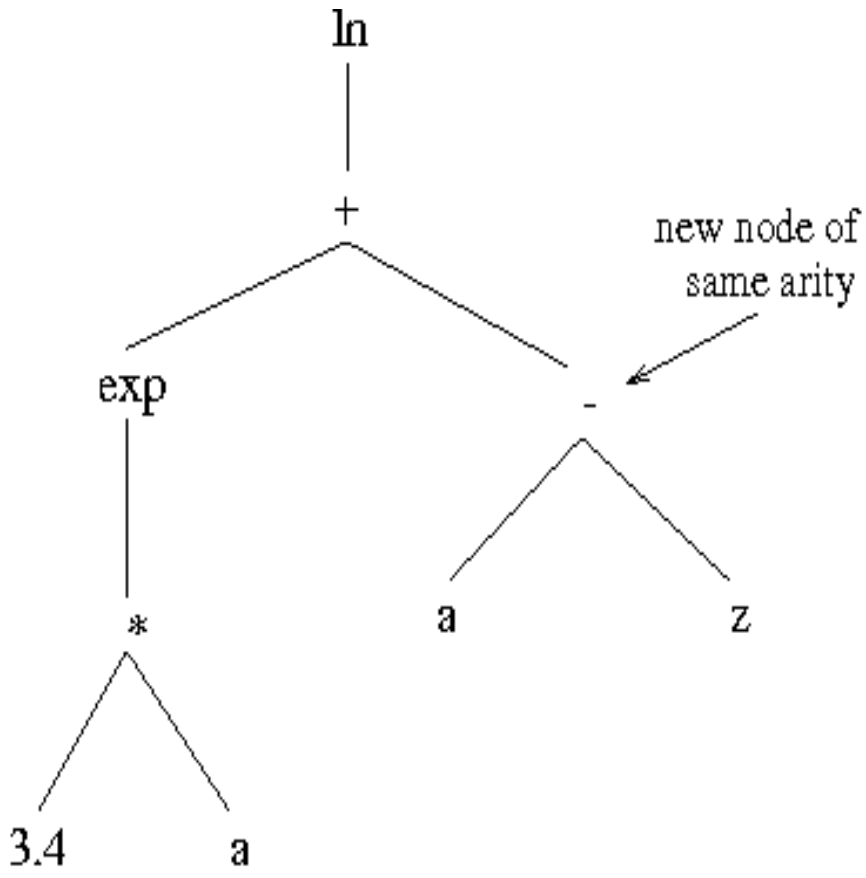
Crossover: Figure 10.3



Mutation: Figure 10.4

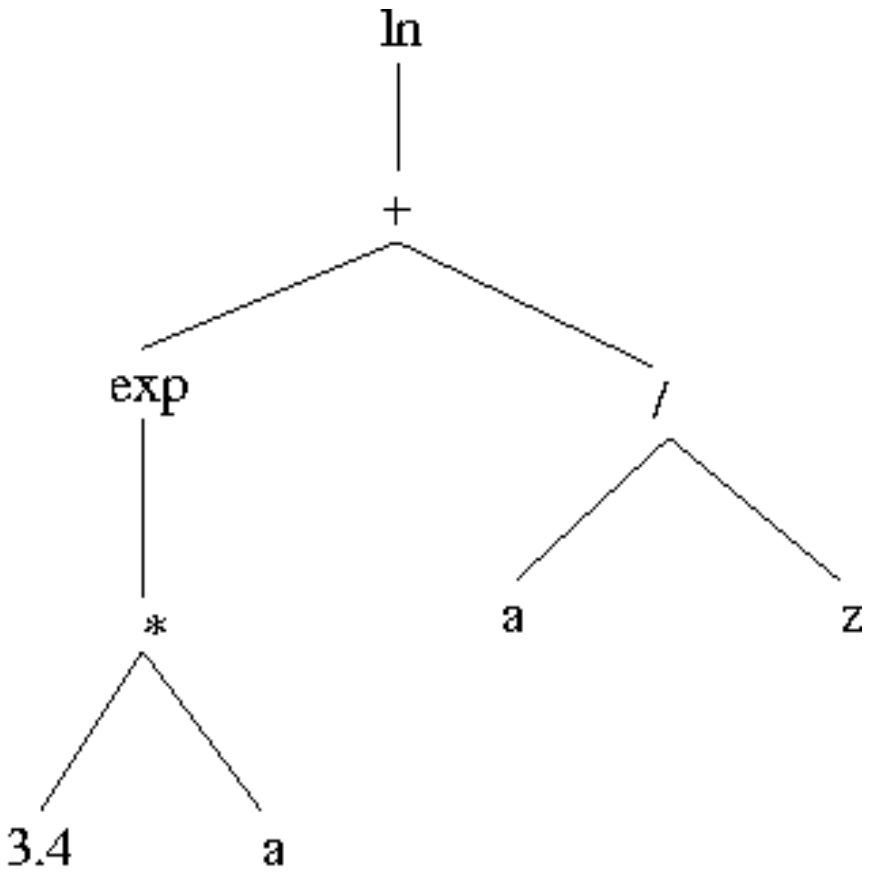


(a) Original Tree

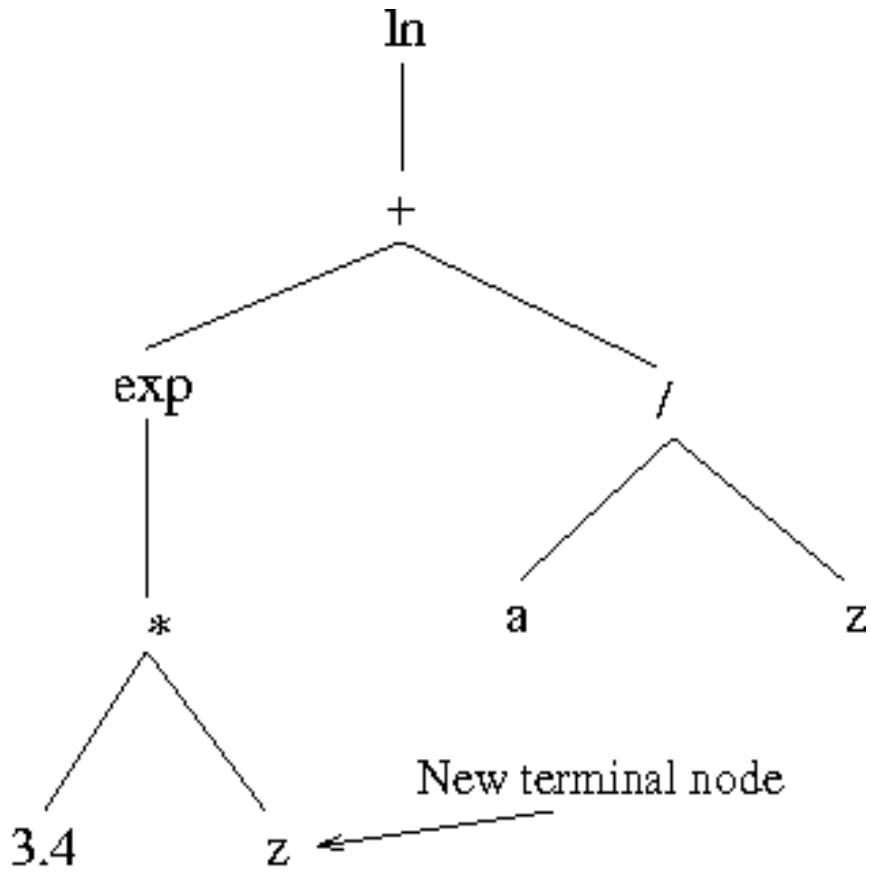


(b) Function Node Mutation

Mutation: Figure 10.4 (cont)

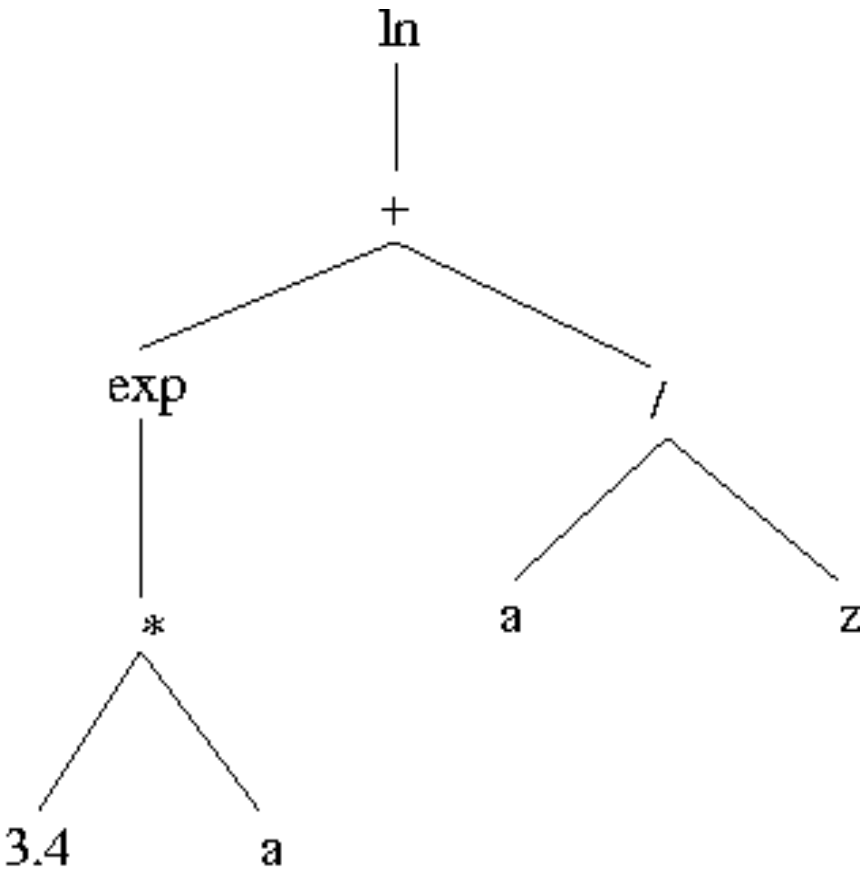


(c) Original Tree

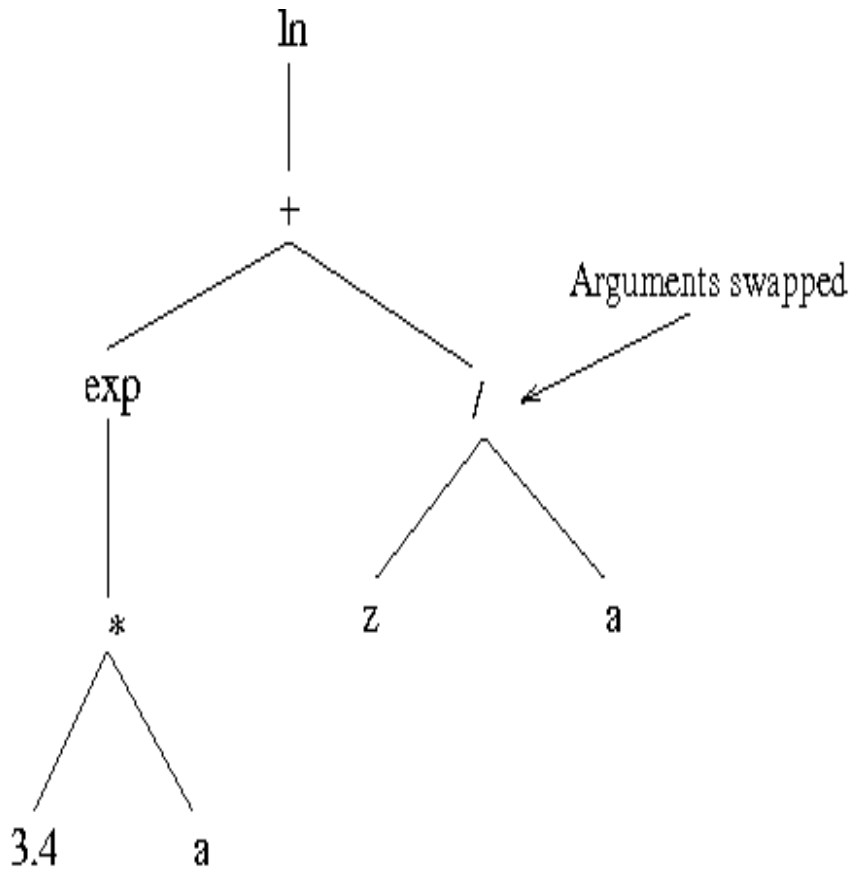


(d) Terminal Node Mutation

Mutation: Figure 10.4 (cont)

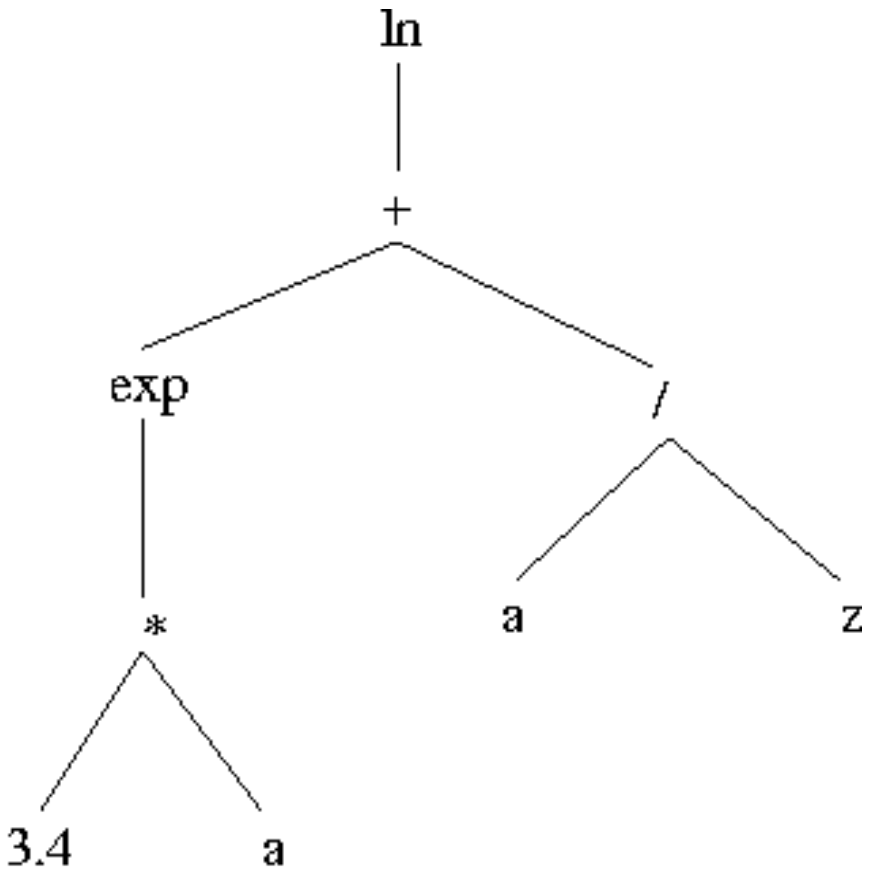


(e) Original Tree

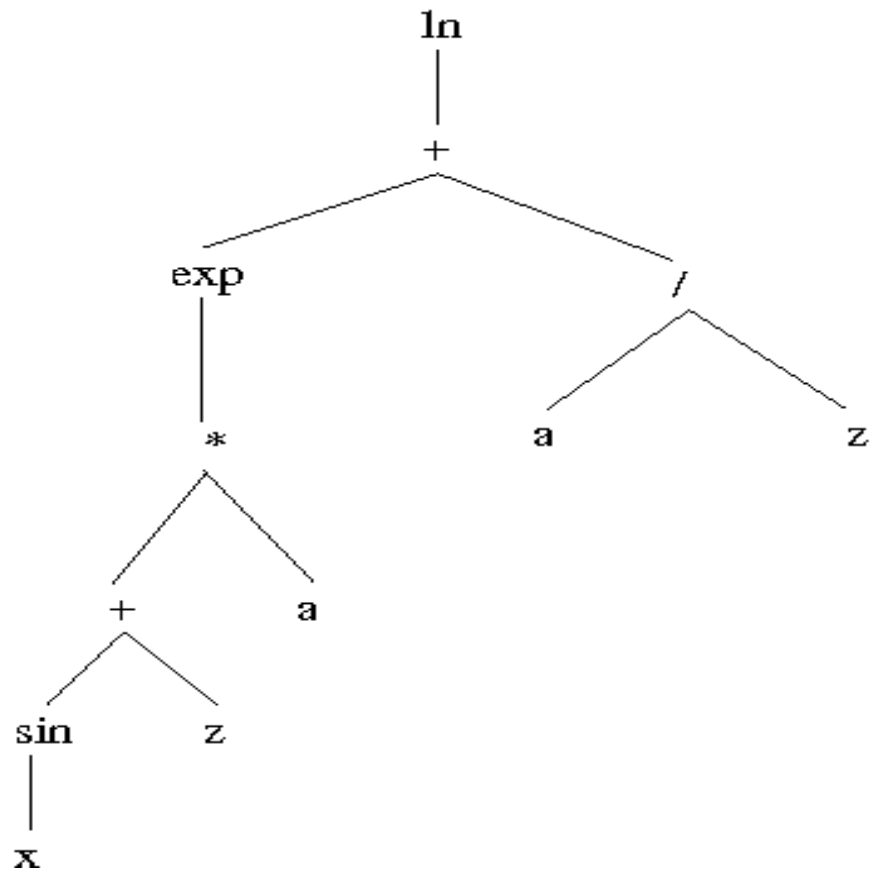


(f) Swapping Mutation

Mutation: Figure 10.4 (cont)

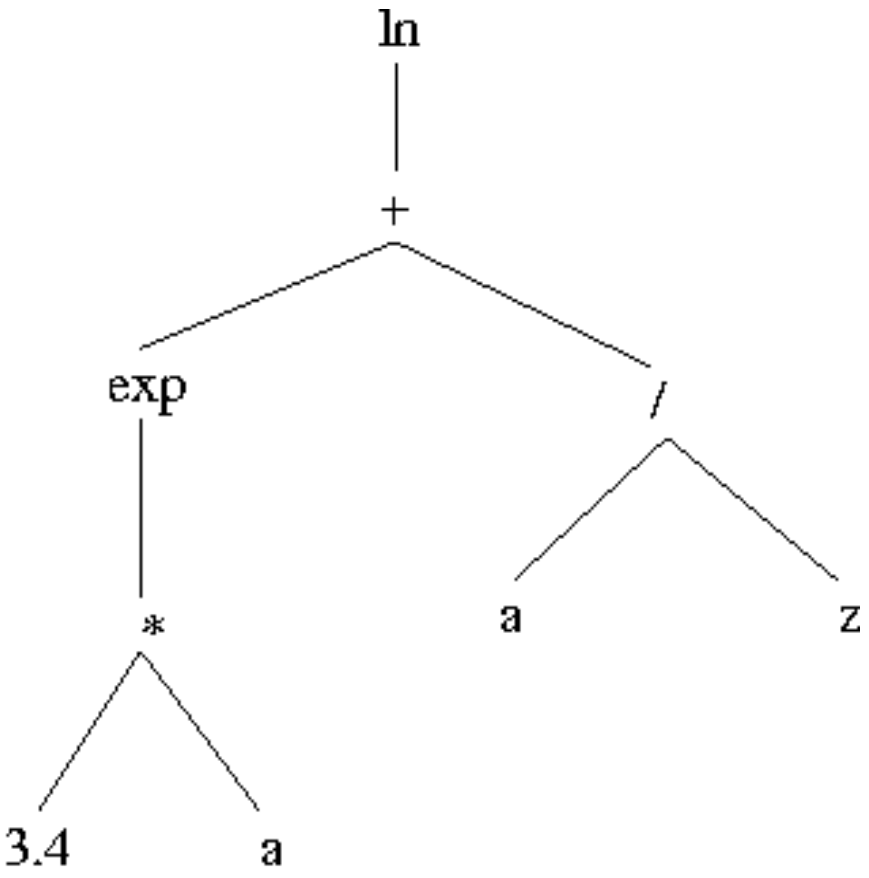


(g) Original Tree

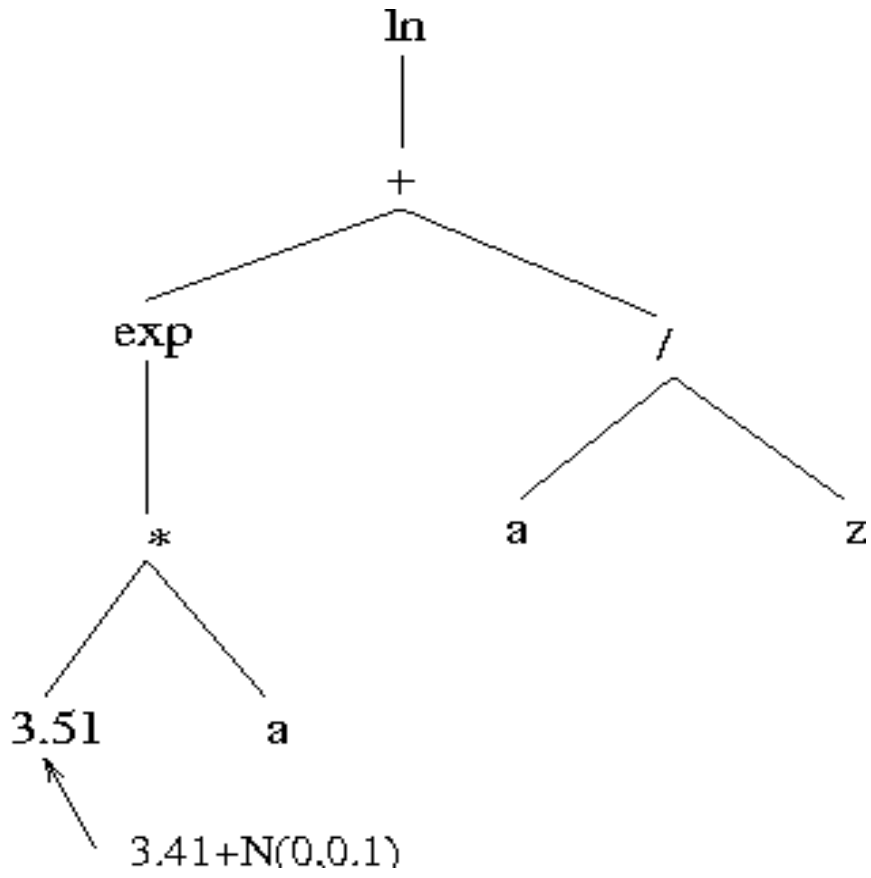


(h) Grow Mutation

Mutation: Figure 10.4 (cont)

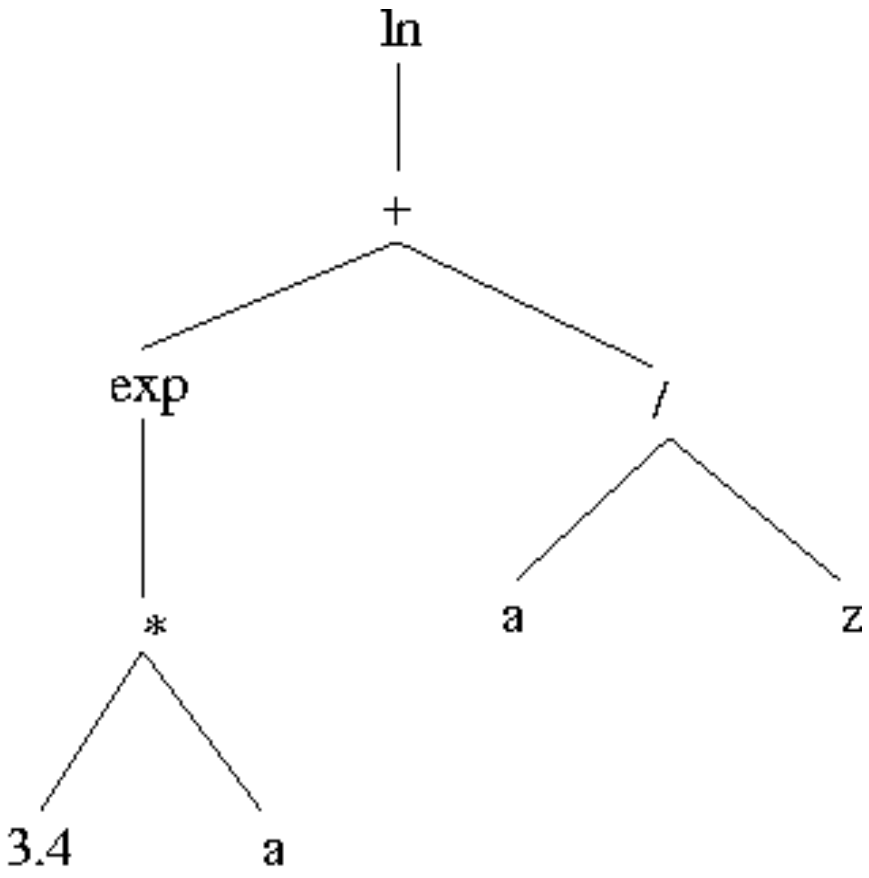


(i) Original Tree

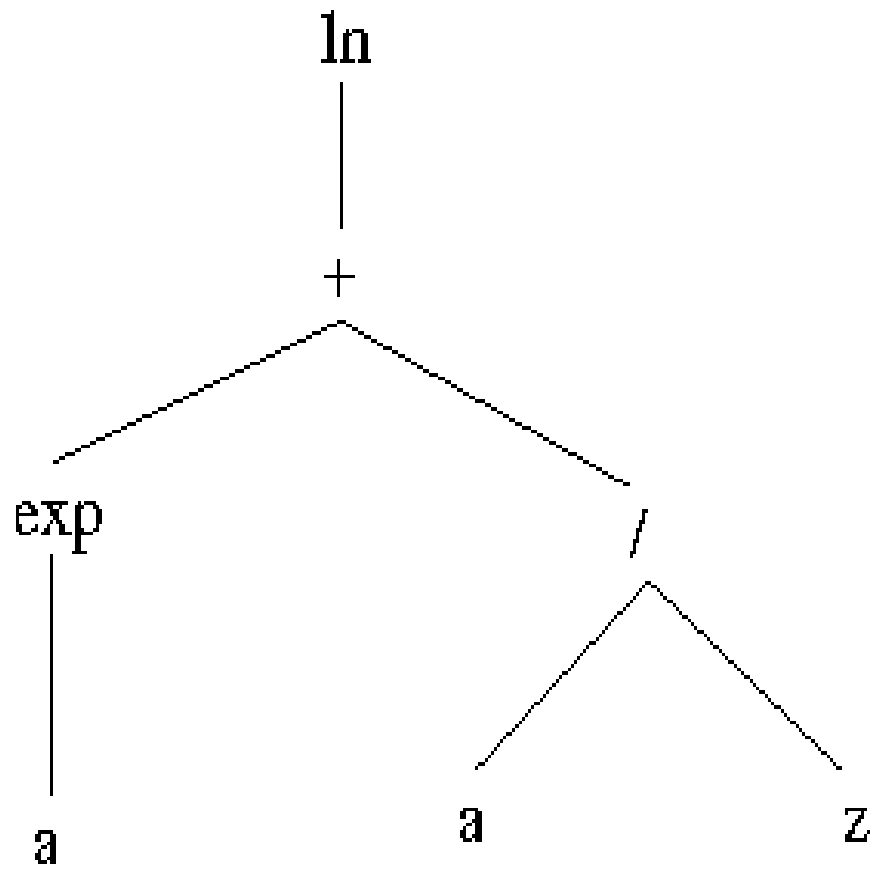


(j) Gaussian Mutation

Mutation: Figure 10.4 (cont)



(k) Original Tree



(l) Trunc Mutation

Building-Block Approach to GP

- Start with the most simple trees, with trees growing during the evolutionary process as needed
- Initial individuals consist of only a root and the immediate children of that node
- Expand trees when
  - the simplicity of the population's individuals can no longer account for the complexity of the problem to be solved
  - no improvement in the fitness of any of the individuals within the population is observed
- Expansion occurs by adding a randomly generated building block (i.e. a new node) to individuals
- Expansion occurs at a specified expansion probability, p_e
- Helps to reduce the computational complexity of the evolution process, and to produce smaller individuals