Computational Intelligence

Unit # 13

Example 3

- Consider an imaginary medical system designed to recommend a dose of quinine to a patient or doctor based on the likelihood that that patient might catch malaria while on vacation.

Fuzzy Reasoning: Example 3

- Average temperature of destination (*T)*
- Average humidity of destination (*H)*
- Proximity to large bodies of water (*P)*
- Industrialization of destination (*I)*
- Dose of Quinine *(Q)*

$$M_{TH}(x) = \begin{cases} \frac{x-25}{75} & for\ x \geq 25 \\ 0 & for\ x < 25 \end{cases}$$

$$M_{HH}(x) = \frac{x}{100}$$

$$M_{TL}(x) = \begin{cases} 1-\frac{x}{75} & for\ x \leq 75 \\ 0 & for\ x > 75 \end{cases}$$

$$M_{HL}(x) = 1-\frac{x}{100}$$

Example 3: Membership Functions

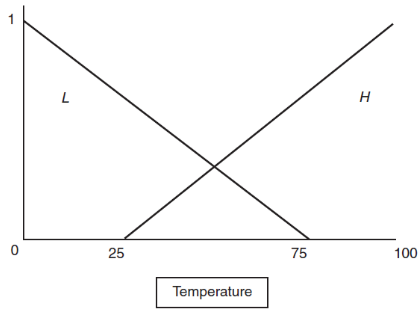
$$M_{PN}(x) = \begin{cases} 1 & for\ x < 10 \\ \frac{40-x}{30} & for\ 10 \leq x < 40 \\ 0 & for\ x \geq 40 \end{cases}$$

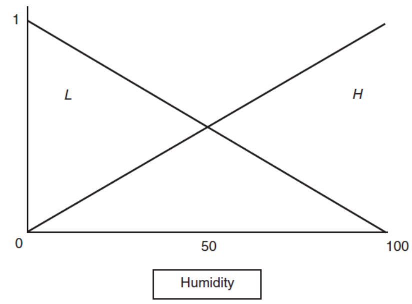
$$M_{IH}(x) = \begin{cases} 0 & for\ x < 10 \\ \frac{x-10}{10} & for\ 10 \leq x < 20 \\ 1 & for\ x \geq 20 \end{cases}$$

$$M_{PF}(x) = \begin{cases} 0 & for\ x < 10 \\ \frac{x-10}{30} & for\ 10 \leq x < 40 \\ 1 & for\ x \geq 40 \end{cases}$$

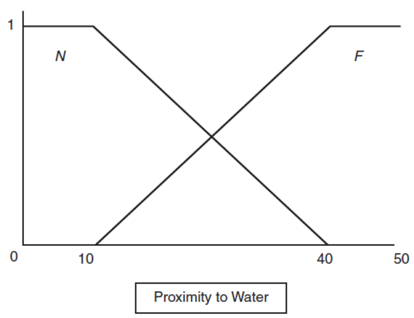
$$M_{IL}(x) = \begin{cases} 1 & for\ x < 10 \\ \frac{20-x}{10} & for\ 10 \leq x < 20 \\ 0 & for\ x \geq 20 \end{cases}$$

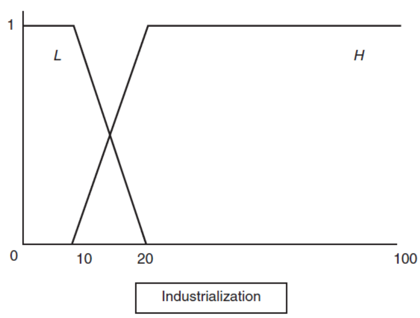
Example 3: Membership Functions (Cont'd)





Example 3: Membership Functions (Cont'd)



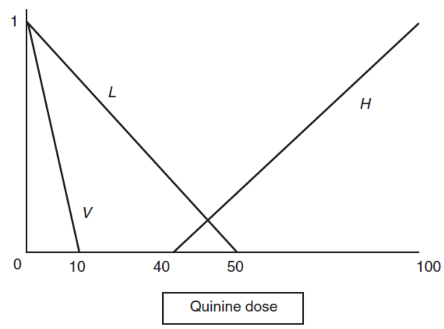


Example 3: Membership Functions (Cont'd)

$$M_{QV}(x) = \begin{cases} \frac{10-x}{10} & for\ x \le 10 \\ 0 & for\ x > 10 \end{cases}$$

$$M_{QL}(x) = \begin{cases} \frac{50-x}{50} & for\ x \le 50 \\ 0 & for\ x > 50 \end{cases}$$

$$M_{QH}(x) = \begin{cases} 0 & for\ x \le 40 \\ \frac{x-40}{60} & for\ x > 40 \end{cases}$$



V: Very Low Dose
L: Low Dose
H: High Dose

Example 3: Rules

- Rule 1
  - IF temperature is high
  - AND humidity is high
  - AND proximity to water is near
  - AND industrialization is low
  - THEN quinine dose is high
- Rule 2
  - IF industrialization is high
  - THEN quinine dose is low

Example 3: Rules (Cont'd)

- Rule 3
  - IF humidity is high
  - AND temperature is high
  - AND (industrialization is low
    OR proximity to water is near)
  - THEN quinine dose is high
- Rule 4
  - IF temperature is low
  - AND humidity is low
  - THEN quinine dose is very low

Example 3: Input Data

- We will examine five sets of data, for five individuals, each of whom is traveling to a country that is at risk from malaria.
- The crisp data are as follows:
  - temperature = {80, 40, 30, 90, 85}
  - humidity = {10, 90, 40, 80, 75}
  - proximity to water = {15, 45, 20, 5, 45}
  - industrialization = {90, 10, 15, 20, 10}
- Hence, for example, person three is traveling to an area where the average temperature is 30, the humidity is 40, the distance to water is 20, and the level of industrialization is 15.

Example 3: Fuzzification

For traveler # 1

- M_{TH} (80) = 0.733
- M_{TL} (80) = 0

- M_{HH} (10) = 0.1
- M_{HL} (10) = 0.9

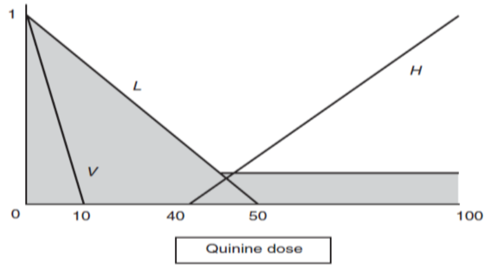
- M_{PN} (15) = 0.833
- M_{PF} (15) = 0.167

- M_{IH} (90) = 1
- M_{IL} (90) = 0

Example 3: Inferencing

- Rule 1: 0
- Rule 2: 1
- Rule 3: 0.1
- Rule 4: 0
- In other words
  - Very low dose (V) = 0
  - Low does (L) = 1
  - High dose (H) = 0.1

Example 3: Defuzzification



Fuzzification for all 5 Cases

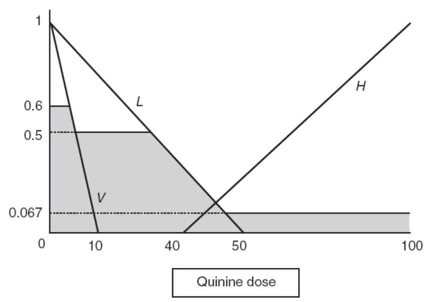
$$M_{TH} = \{0.733, 0.2, 0.067, 0.867, 0.8\}$$
$$M_{TL} = \{0, 0.467, 0.6, 0, 0\}$$
$$M_{HH} = \{0.1, 0.9, 0.4, 0.8, 0.75\}$$
$$M_{HL} = \{0.9, 0.1, 0.6, 0.2, 0.25\}$$
$$M_{PN} = \{0.833, 0, 0.667, 1, 0\}$$
$$M_{PF} = \{0.167, 1, 0.333, 0, 1\}$$
$$M_{IH} = \{1, 0, 0.5, 1, 0\}$$
$$M_{IL} = \{0, 1, 0.5, 0, 1\}$$

Inferencing for all 5 Cases

- Rule 1
  - (high dose): {0, 0, 0.067, 0, 0}
- Rule 2
  - (low dose): {1, 0, 0.5, 1, 0}
- Rule 3
  - (high dose): {0.1, 0.2, 0.067, 0.8, 0.75}
- Rule 4
  - (very low dose): {0, 0.1, 0.6, 0, 0}
- Since both Rule 1 and 3 suggest high dose, we can max of them. Thus
  - high dose: {0.1, 0.2, 0.067, 0.8, 0.75}

Example 3 (Cont'd)

- For Traveler 3
  - V: 0.6
  - L: 0.5
  - H: 0.067



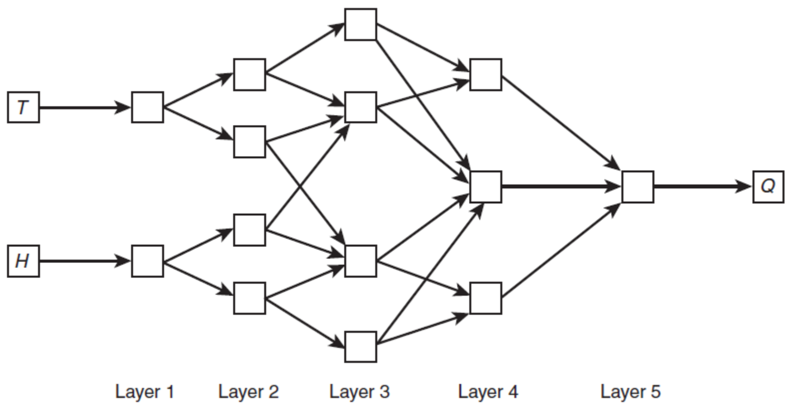
Neuro-Fuzzy Systems

- A **neuro-fuzzy** system is a neural network that learns to classify data using fuzzy rules and fuzzy classifications (fuzzy sets).
- A neuro-fuzzy system has advantages over fuzzy systems and traditional neural networks:
  - A traditional neural network is often described as being like a "black box," in the sense that once it is trained, it is very hard to see why it gives a particular response to a set of inputs.
  - This can be a disadvantage when neural networks are used in mission-critical tasks where it is important to know why a component fails.
  - Fuzzy systems and neuro-fuzzy do not have this disadvantage.

Architecture of a Fuzzy Neural Network

- Typically, a fuzzy neural network is a five-layer feed-forward network.
  - Input layer – receives crisp inputs
  - Fuzzy input membership functions
  - Fuzzy rules
  - Fuzzy output membership functions
  - Output layer – output crisp values

Typical Five-Layer Neuro-Fuzzy Network



Layers Description

- The first layer simply passes its crisp input to the second layer.
- The second layer contains information about the various fuzzy sets that are being used to map the crisp inputs.
- Typically the neurons used in this second layer have triangulation functions, which represent the triangular membership functions of the fuzzy sets, although any functions can be used.

Layers Description (Cont'd)

- The third layer represents the fuzzy rules of the system. Each neuron in this layer represents a single fuzzy rule.
- Typically, the system would be setup with initial fuzzy rules built in, and the network would develop suitable weightings to give the best possible responses.
- The fourth layer in the network contains the neurons that represent the membership functions of the various possible outputs of the fuzzy rules.

Layers Description (Cont'd)

- The fifth and final layer is the layer that combines and defuzzifies the various outputs to produce one single crisp output for the network in response to a given set of inputs.
- The connection between the layers have weights associated with them, and using the methods such as backpropagation or evolutionary algorithms, the system is able to learn.

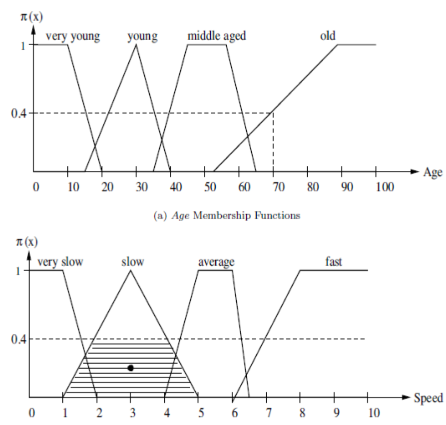
Evolutionary Neuro-Fuzzy Systems

How the System Learns

- The neuro-fuzzy system learns using the same technique used by traditional neural networks.
- Learning is done by adjusting the weights of the connections between neurons in the network or by adjusting the boundaries of the fuzzy membership functions.
- Neuro-Fuzzy systems are very robust and can ususally detects rules that have been entered erroneously.
- For example, if one expert give the following rule:
  - If T_H and H_H then Q_H
- And another expert gave the following rule
  - If T_H and H_H then Q_V
- The system would find the incorrect rule by setting all the weights for the wrong rule to zero.

Example 1 from Unit # 12

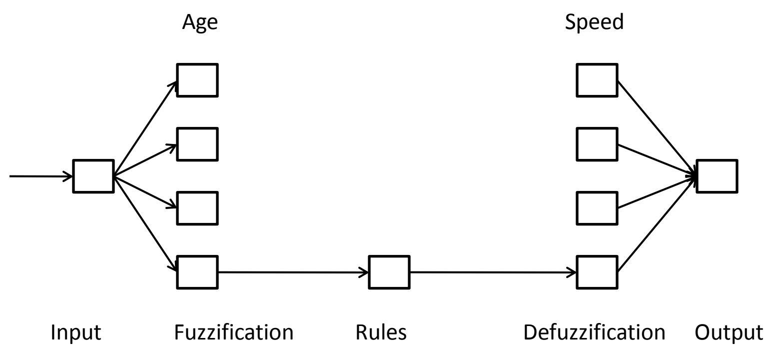
- Rule
  - If *Age* is *Old* the *Speed* is *Slow*
- What can be said about *Speed* if *Age* has the value of 70?



(a) *Age* Membership Functions

(b) *Speed* Membership Functions

Example 1: Neuro-Fuzzy Network



Example 2 from Unit # 2

Let us suppose that we are designing a simple braking system for a car, which is designed to cope when the roads are icy and the wheels lock.

The rules for our system might be as follows:

Rule 1 IF pressure on brake pedal is medium
THEN apply the brake

Rule 2 IF pressure on brake pedal is high
AND car speed is fast
AND wheel speed is fast
THEN apply the brake

Rule 3 IF pressure on brake pedal is high
AND car speed is fast
AND wheel speed is slow
THEN release the brake

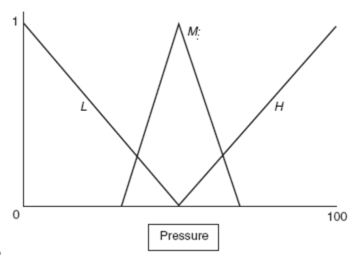
Rule 4 IF pressure on brake pedal is low
THEN release the brake

For this simple example, we will assume that brake pressure is measured from 0 (no pressure) to 100 (brake fully applied). We will define brake pressure as having three linguistic values: high (H), medium (M), and low (L), which we will define as follows:

$H = \{(50, 0), (100, 1)\}$

$M = \{(30, 0), (50, 1), (70, 0)\}$

$L = \{(0, 1), (50, 0)\}$

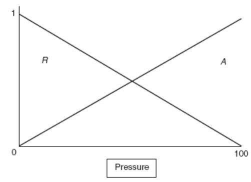


Example 2 (Cont'd)

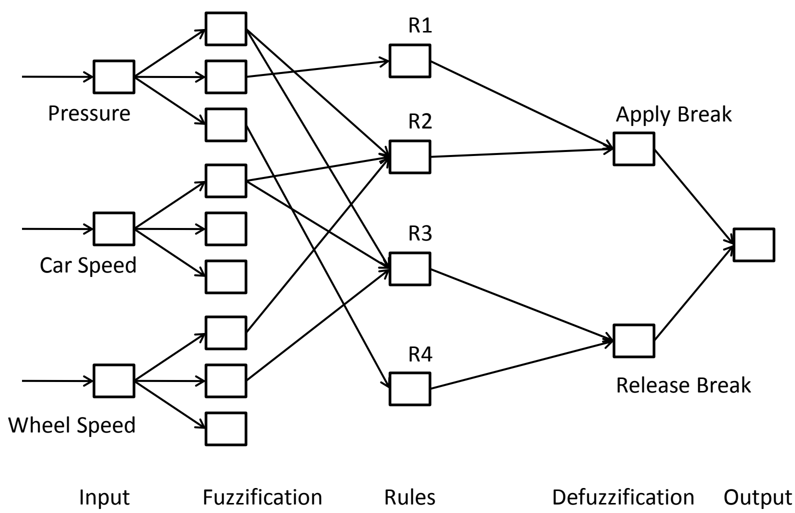
Similarly, we must consider the wheel speed. We will define the wheel speed as also having three linguistic values: slow, medium, and fast. We will define the membership functions for these values for a universe of discourse of values from 0 to 100:

$S = \{(0, 1), (60, 0)\}$
$M = \{(20, 0), (50, 1), (80, 0)\}$
$F = \{(40, 0), (100, 1)\}$

For the sake of simplicity, we will define the linguistic variable *car speed* using the same linguistic values (S, M, and F for slow, medium, and fast), using the same membership functions. Clearly, in a real system, the two would be entirely independent of each other.



Example 2: Neuro-Fuzzy Network



Genetic Programming

Genetic Programming

- Genetic programming is the youngest member of the evolutionary algorithm family.
- Besides the particular representation (using trees as chromosomes), it differs from other EA strands in its application area.
- While the EAs discussed so far are typically applied to optimization problems, GP could instead be positioned in machine learning.
- GP is typically used to seek models with maximum fit.

Sketch of Genetic Programming

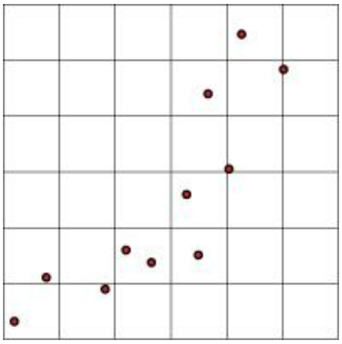
Representation	Tree Structures
Recombination	Exchange of Subtrees
Mutation	Random change in trees
Parent Selection	Fitness Proportional
Survivor Selection	Generational Replacement

Implication of Tree-based Representations

- Contrary to GAs where the size of individuals are usually fixed, a GP population will usually have individuals of different size, shape and complexity.
- Here size refers to the tree depth, and shape refers to the branching factor of nodes in the tree.
- The size and shape of a specific individual are also not fixed, but may change due to application of the reproduction operators.

Symbolic Regression

- Applying GP to symbolic regression, we could simply input a set of data points and get a function that fits the data appropriately, such as f(x) = $x^3 + 2x^2 - x + 1$.



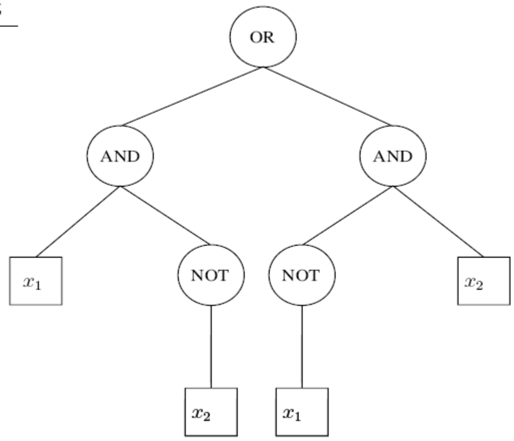
Grammar

- A grammar forms an important part of chromosome representation.
- As part of the grammar, a terminal set, function set, and semantic rules need to be defined.
- The terminal set specifies all the variables and constants, while the function set contains all the functions that can be applied to the elements of the terminal set.
- These functions may include mathematical, arithmetic and/or Boolean functions.

Example 1

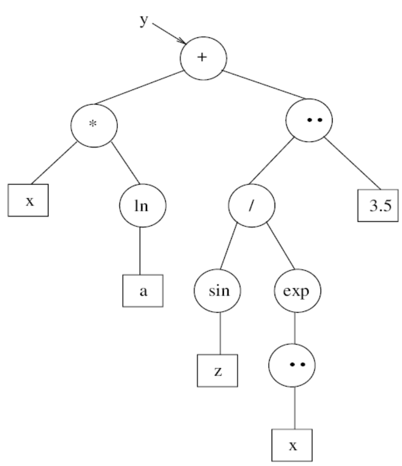
x_1	x_2	Target Output
0	0	0
0	1	1
1	0	1
1	1	0

- For this problem, the function set is defined as *{AND,OR,NOT}, and* the terminal set is *{x1, x2} where x1, x2 ∈{0, 1}.*



Example 2

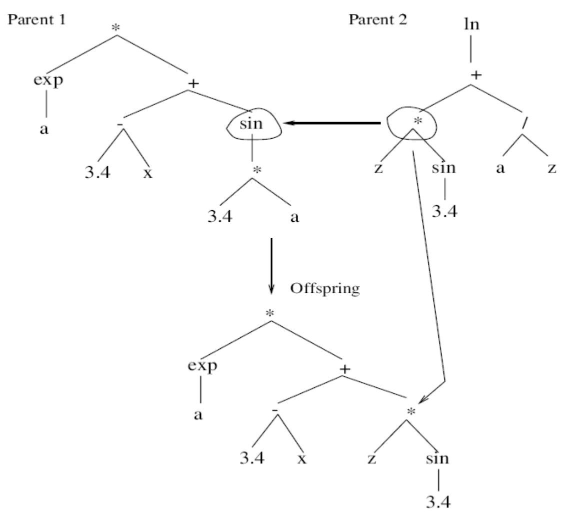
- y:=x*ln(a)+sin(z)/exp (-x)-3.5;
- The terminal set is specified as *{a, x, z, 3.5} with a, x, z ∈R.*
- *The minimal function set* is given as *{−,+, *, /, sin, exp, ln}.*



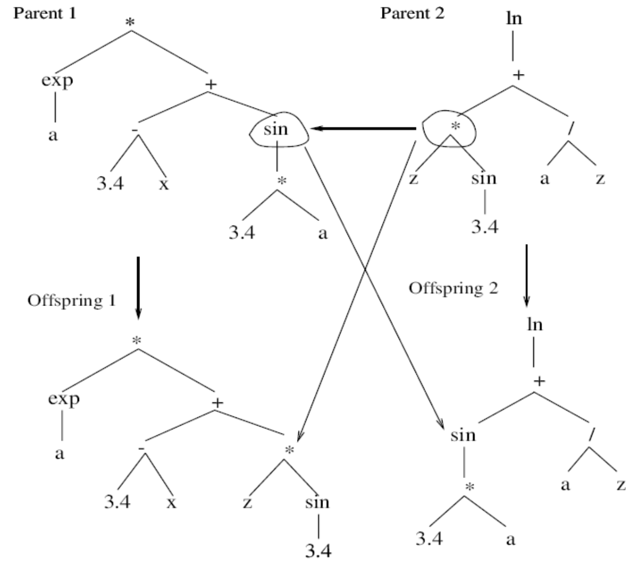
Initialization

- The initial population is generated randomly 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.
- The branching factor (the number of children) of the root, and each non-terminal node, are determined by the arity of the selected function.
- For each non-root node, the initialization algorithm randomly selects an element either from the terminal set or the function set.
- As soon as an element from the terminal set is selected, the corresponding node becomes a leaf node and is no longer considered for expansion.

Crossover (Creating 1 Offspring)



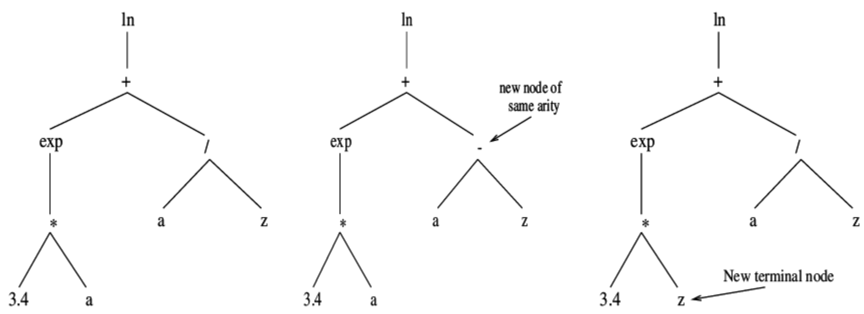
Crossover (Creating 2 Offspring)



Mutation

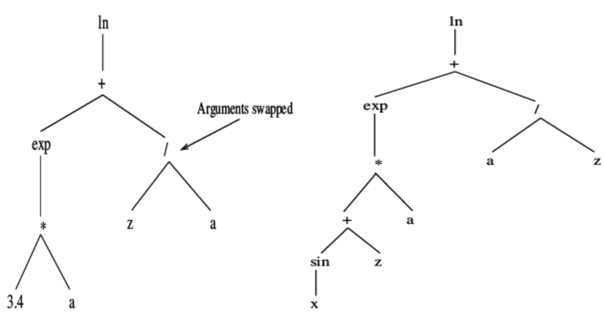
- **Function node mutation:** A non-terminal node, or function node, is randomly selected and replaced with a node of the same arity, randomly selected from the function set.
- ***Terminal node mutation:*** *A leaf node, or terminal node, is randomly selected* and replaced with a new terminal node, also randomly selected from the terminal set.
- ***Swap mutation:*** *A function node is randomly selected and the arguments of* that node are swapped.
- ***Grow mutation:*** *With grow mutation a node is randomly selected and replaced* by a randomly generated subtree. The new subtree is restricted by a predetermined depth.

Mutation (Cont'd)



(a) Original tree (b) Function node (c) Terminal node

Mutation (Cont'd)



(d) Swapping (e) Grow

Recap

- Evolutionary Algorithms
  - Evolution Strategies, Evolutionary Programming, Genetic Algorithms, Genetic Programming
- Swarm Intelligence
  - Ant Colony Optimization, Particle Swarm Optimization, Artificial Bee Colony
- Other Meta-Heuristics
  - Artificial Immune Systems, Tabu Search, Simulated Annealing
- Neural Networks
- Fuzzy Logic
- Hybrid Systems
  - Evolutionary Neural Networks, Neuro Fuzzy Systems