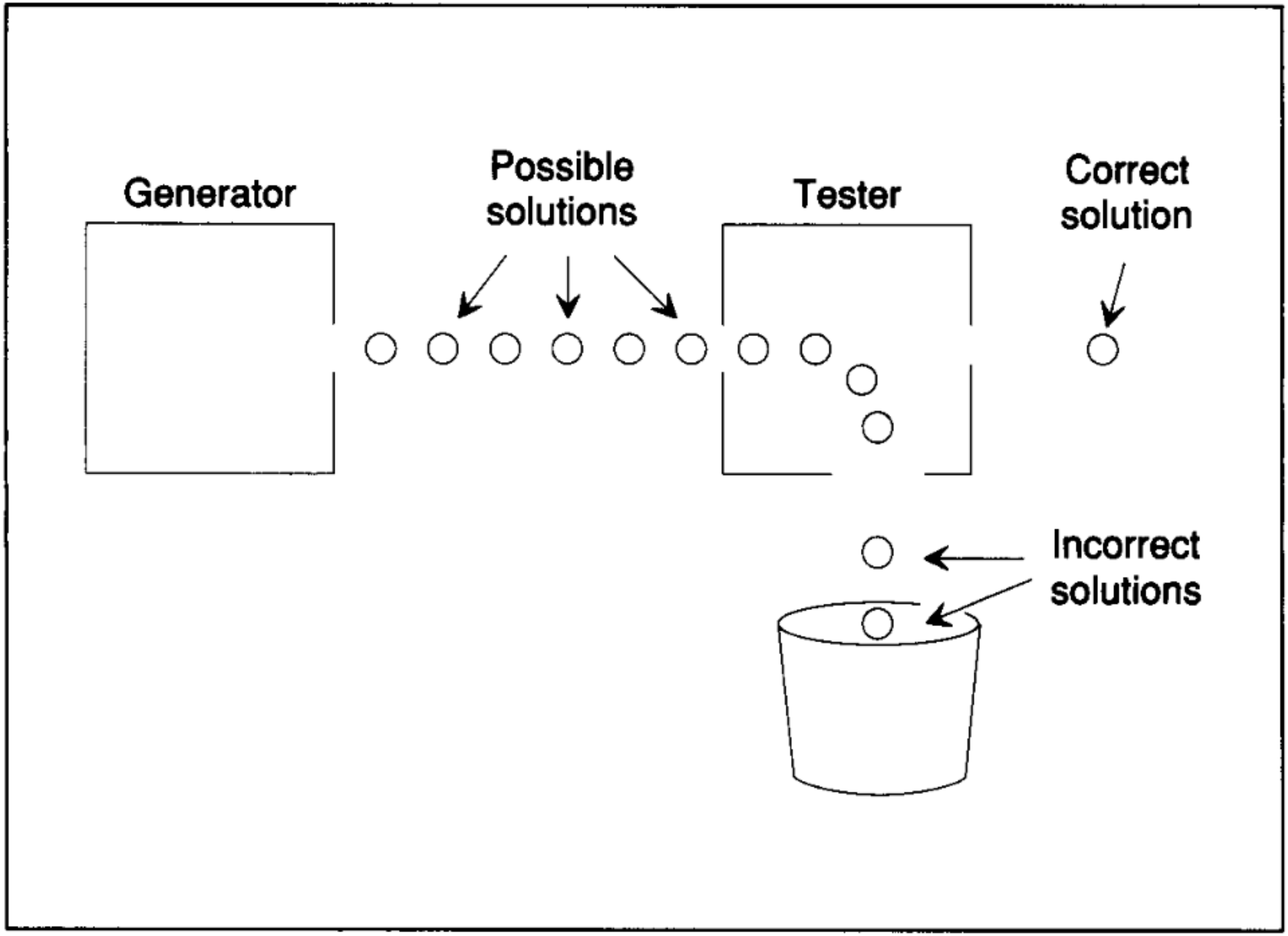
Applied Artificial Intelligence

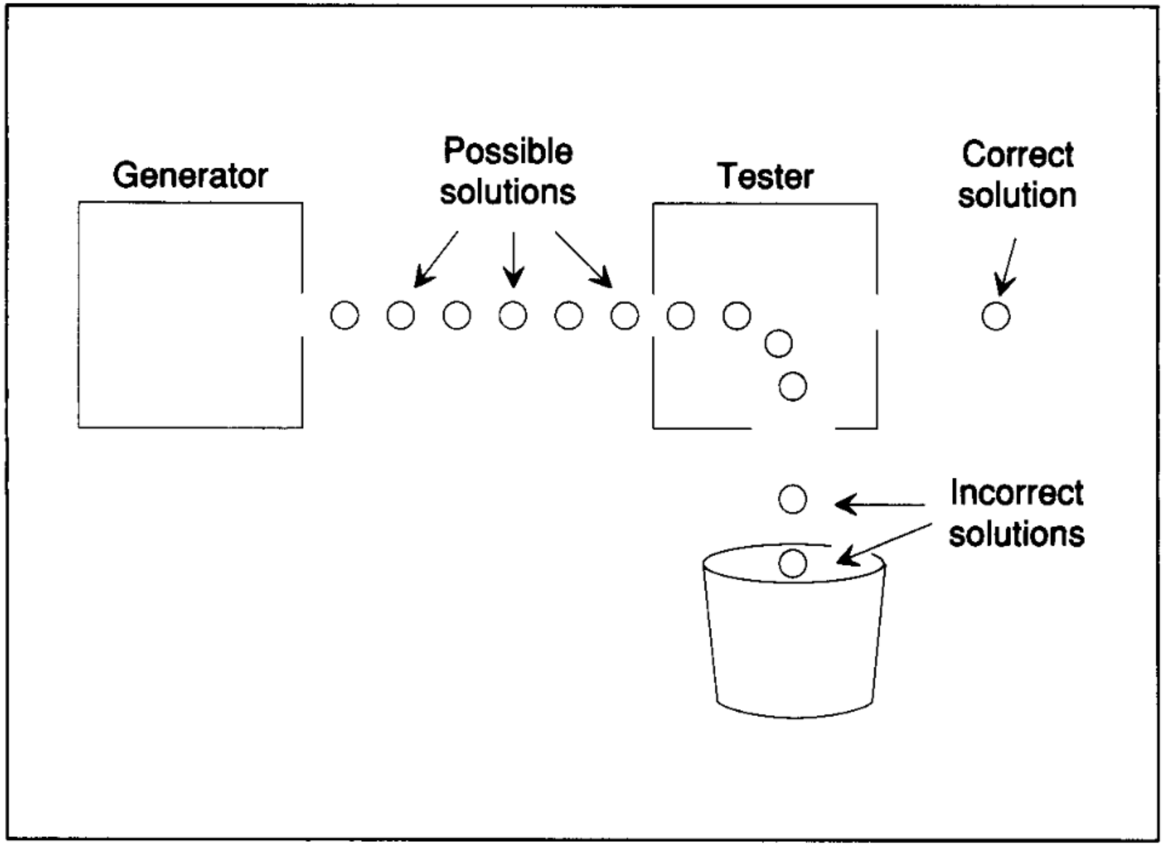
Session 5: Informed Searching as an AI Method

Fall 2018
NC State University
Instructor: Dr. Behnam Kia
Course Website: https://appliedai.wordpress.ncsu.edu/



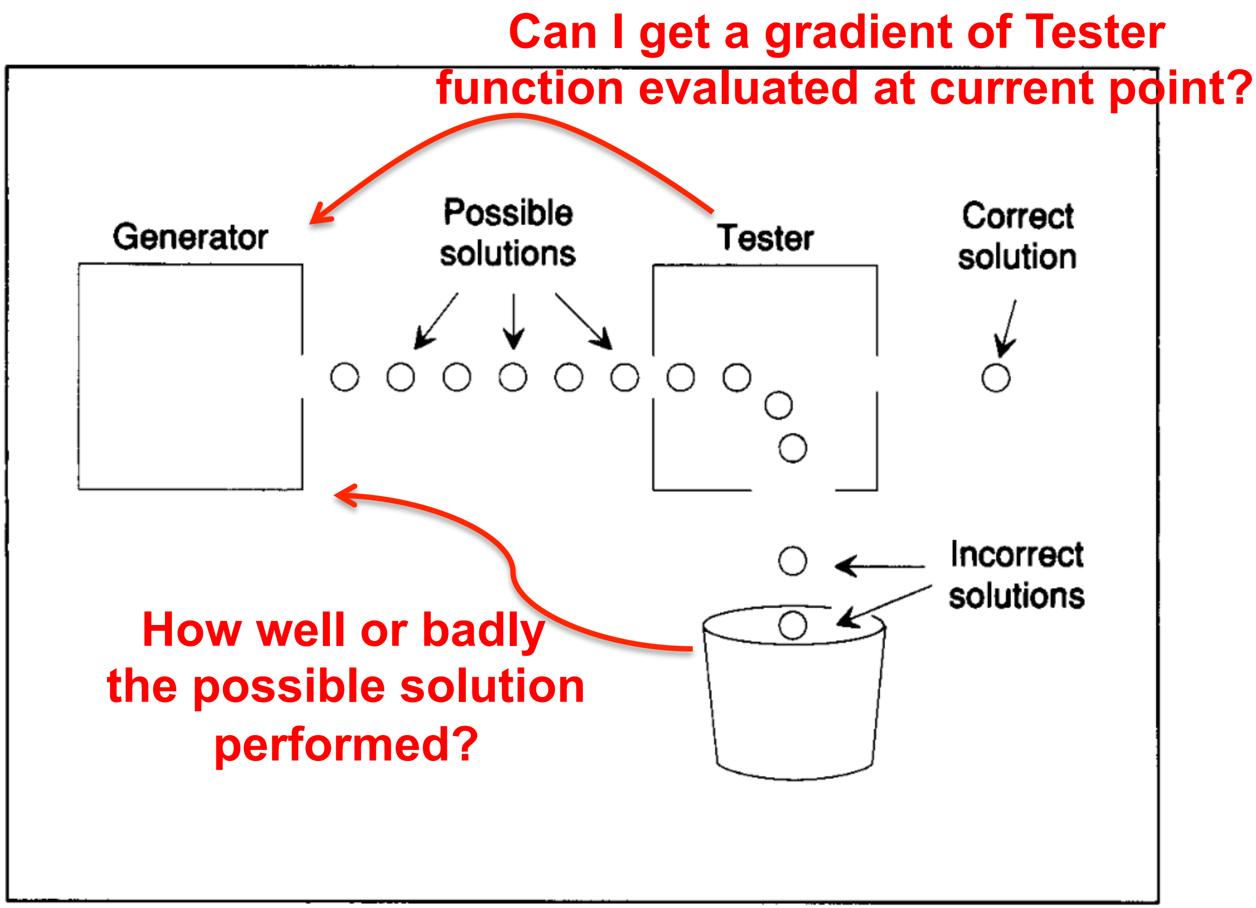
Picture from Artificial Intelligence, Patrick Winston

Uninformed, Blind Search



Picture from Artificial Intelligence, Patrick Winston

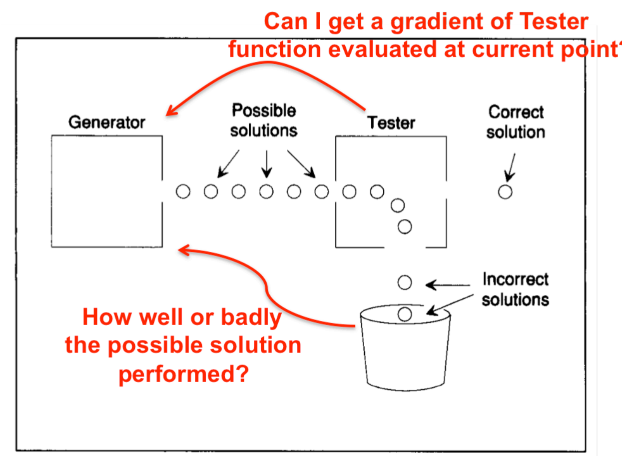
Informed Search



Picture from Artificial Intelligence, Patrick Winston
Red Arrow added by me. – Behnam Kia

Metaheuristic Methods for Global Optimization (Informed Search)

- **Evolutionary Computation**: Inspired by evolutionary biology, creates and evolves a population of solutions that hopefully converge to the global minimum.
- **Simulated Annealing (SA)**: Inspired by statistical physics, SA solution can escape from a local minimum and hopefully converge to the global minimum point.
- **(Stochastic) Gradient Methods**: Will be studying in future sessions.



Picture from Artificial Intelligence, Patrick Winston
Red Arrow added by me. – Behnam Kia

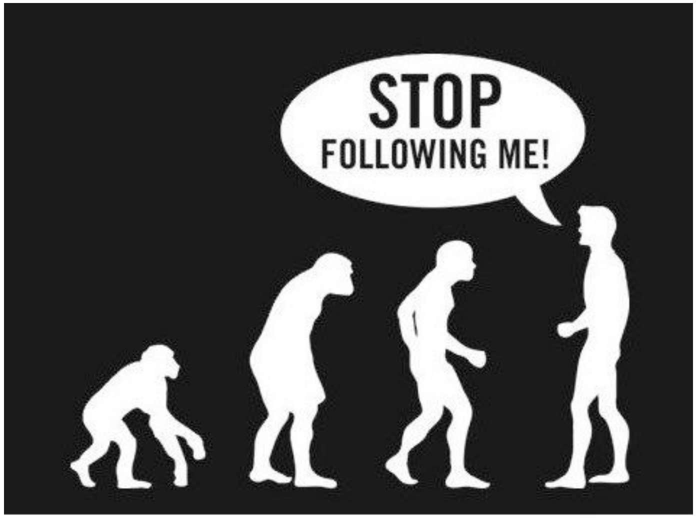
Evolutionary Computation

- A family of algorithms for global optimization inspired by biological evolution.

- Here we study Genetic Algorithm, a popular example from the family of evolutionary computation algorithms.

Biological Evolution

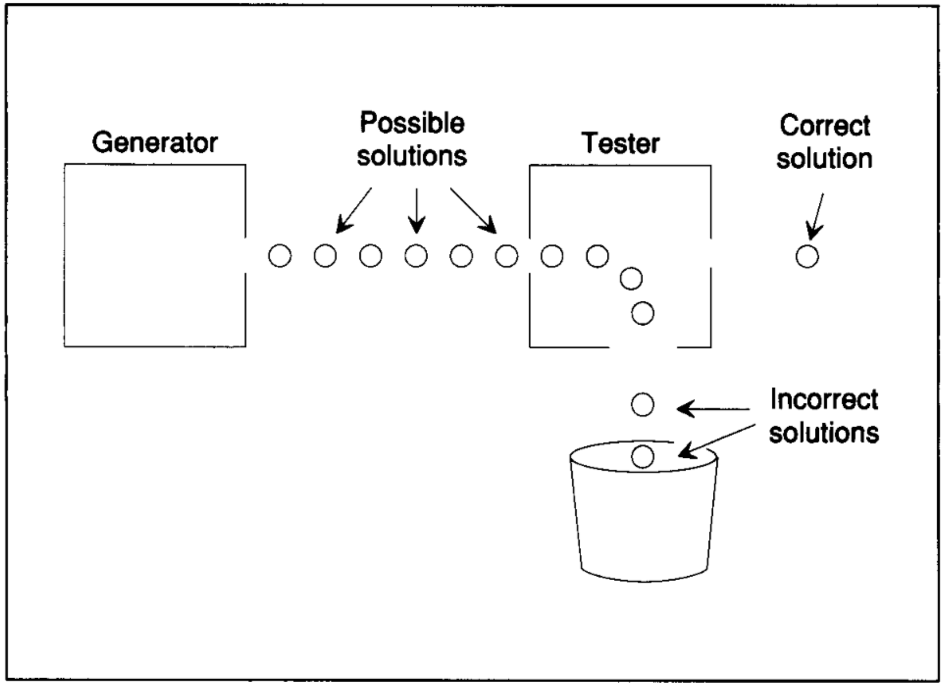
- In a population, more fitted individuals are more likely to survive and reproduce.
- The next generation can inherit the good genes of the parents.
- And at each reproduction, there is some mutation, giving new features to the offspring.
- After many generations, the population can become very fitted and optimal.



www.UShumor.com

Biological Evolution

- In a population, more fitted individuals are more likely to survive and reproduce.
- The next generation can inherit the good genes of the parents.
- And at each reproduction, there is some mutation, giving new features to the offspring.
- After many generations, the population can become very fitted and optimal.



Genetic Algorithm

1. Create an initial population of possible solutions
2. More fitted individuals (according to a fitness function) create the next generation of population.
3. Induce some mutations to the new population.
4. Go to step 2 and repeat until the optimal solutions is found.

Genetic Algorithm: Component

1. Each possible solution is an individual.

   If the problem is *k*-dimensional, (has *k* variables), each individual would have *k* chromosomes.

An Individual=a possible solution
Chromosome=variable in the solution

Genetic Algorithm: Component

2. Initial population (of size *n*):
   - Create *n* individuals with randomly initialized chromosomes.

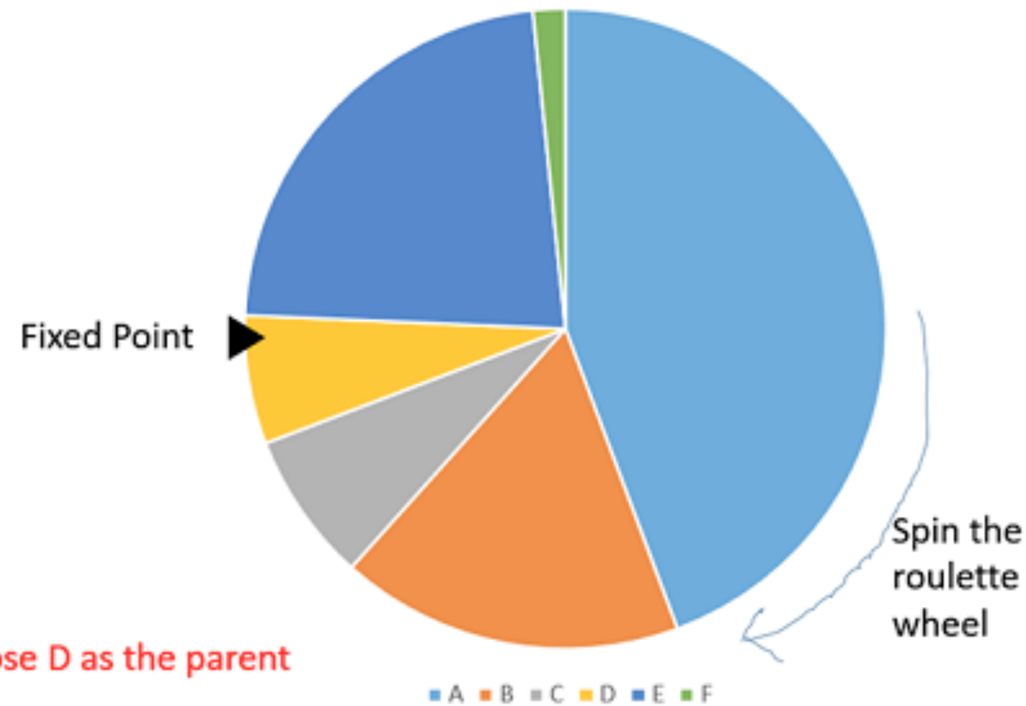
Chromosome=variable in the solution

Genetic Algorithm: Component

3. Evaluating fitness of an individual:
   - Use the cost function

Genetic Algorithm: Component

3. Selection of parents to reproduce:
   - Roulette wheel selection



Chromosome	Fitness Value
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

Choose D as the parent

Picture from https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_parent_selection.htm

Genetic Algorithm: Component

3. Selection of parents to reproduce:
   – Roulette wheel selection
   – Tournament election: With uniform sampling, choose *m* individual at random. Take the most fitted one.

- If an evolutionary computation requires two parents for reproduction, repeat the process above twice, once for each parent.

Genetic Algorithm: Component

4. Crossover:
   - Many different methods, here for continuous chromosomes we use:

$$O = \beta \times p_1 + (1 - \beta) \times p_2$$

O, p_1, and p_2 are the chromosomes of the offspring, parent 1 and parent 2 respectively. β is a random number between 0 and 1. If the individual has k chromosomes, we will have k of above.

Genetic Algorithm: Component

4. Mutation:
   - Many different methods, here for continuous chromosomes we use:

$$O = O + \varepsilon N(0,1)$$

O is the chromosome of the offspring, and we add Gaussian noise to it. ε is the noise scale. If the individual has *k* chromosomes, we will have *k* of above

Genetic Algorithm: Component

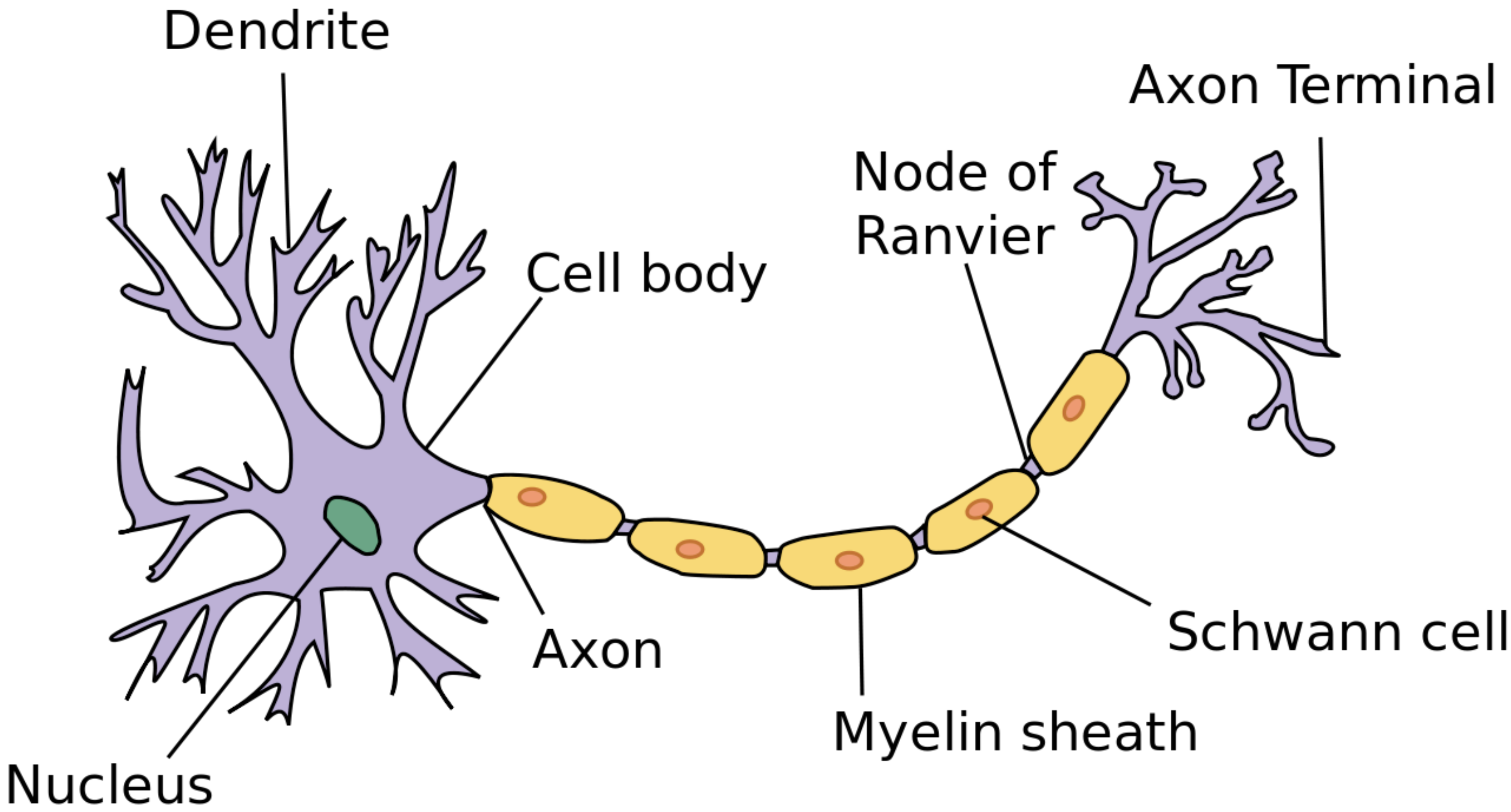
5. Replacement of generations:

   We replace the old generation with the new. Some versions let fitted individuals of the old gen live as well.

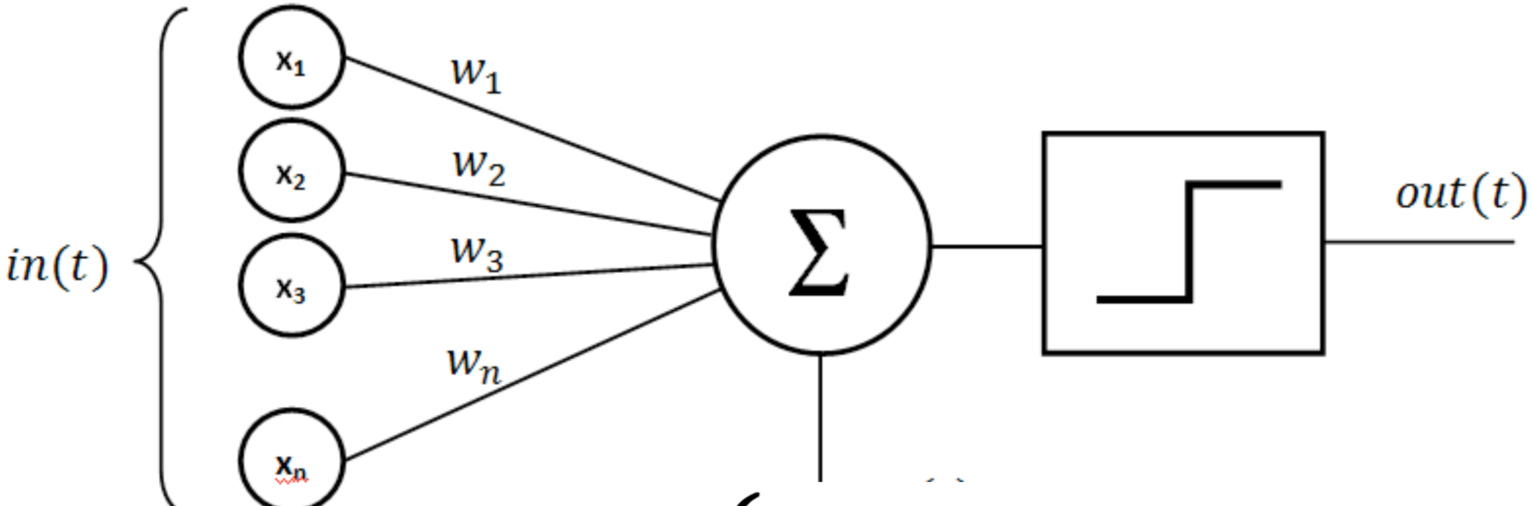
Perceptron (A single Neuron Model)

Perceptron: A Computational Neuron Model

Biological Neuron



Perceptron: A Computational Neuron Model



$$f(x) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

Perceptron with Two Inputs

$$o = \begin{cases} 1 & w_0 x_0 + w_1 x_1 + b > 0 \\ 0 & otherwise \end{cases}$$

Perceptron with Two Inputs

$$o = \begin{cases} 1 & w_0 x_0 + w_1 x_1 + b > 0 \\ 0 & otherwise \end{cases}$$

Divide by w_1

$$o = \begin{cases} 1 & \theta x_0 + x_1 + c > 0 \\ 0 & otherwise \end{cases}$$

Perceptron with Two Inputs

$$o = \begin{cases} 1 & w_0 x_0 + w_1 x_1 + b > 0 \\ 0 & otherwise \end{cases}$$

Divide by w_1

$$o = \begin{cases} 1 & \theta x_0 + x_1 + c > 0 \\ 0 & otherwise \end{cases}$$