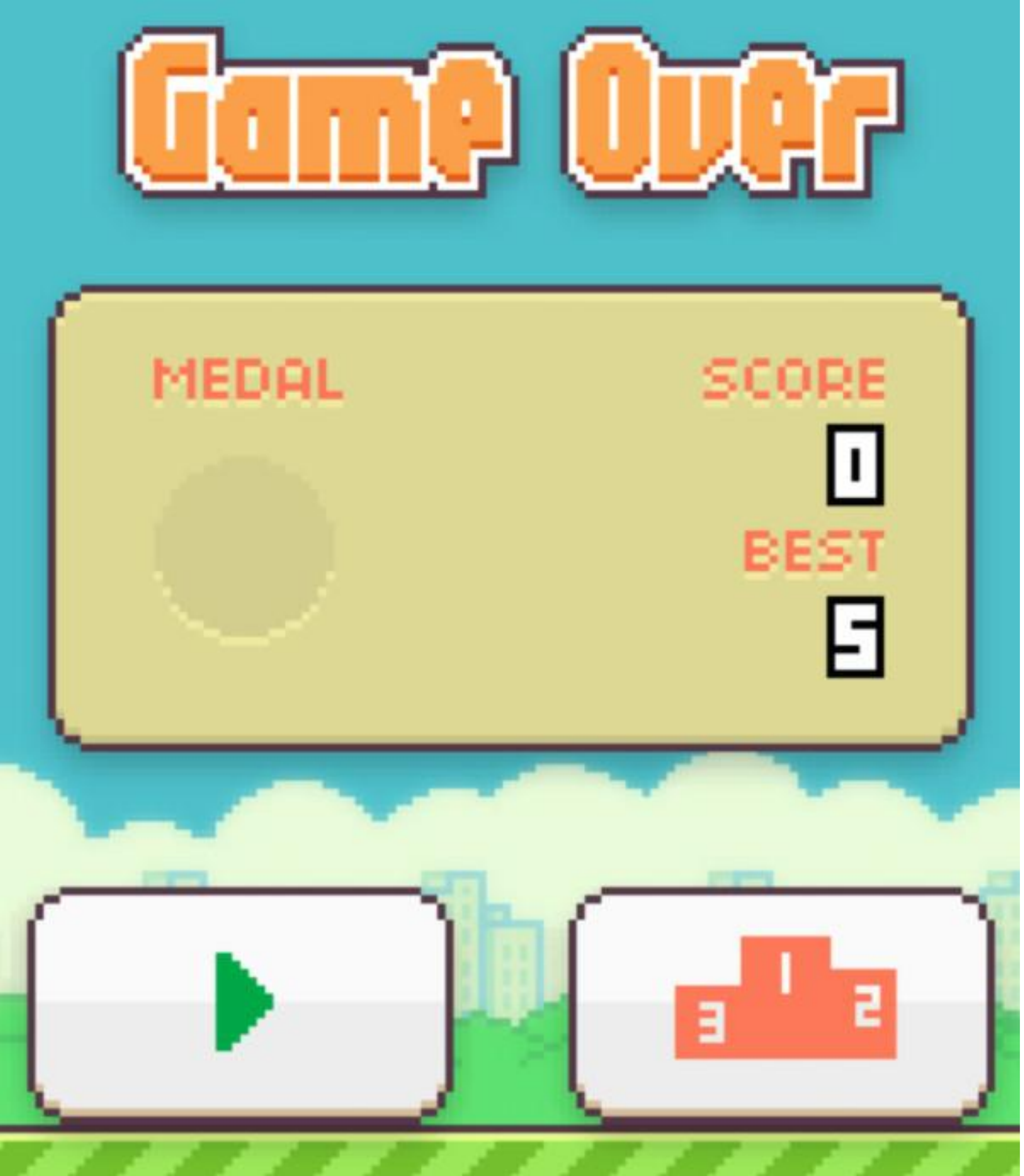
Flappy Birds

using NEAT

Genetics Algorithm

Team 2: Kevin Pang, Humza Butt, Rianna He, Wayne Li, Stella Moon



What is your best score?

What if we can make the flappy bird to **teach itself** to learn how to play the game?

Agenda

- Background
- Overview of Algorithms
- Applications of Algorithms
- Experiment on Flappy Birds
- Conclusion

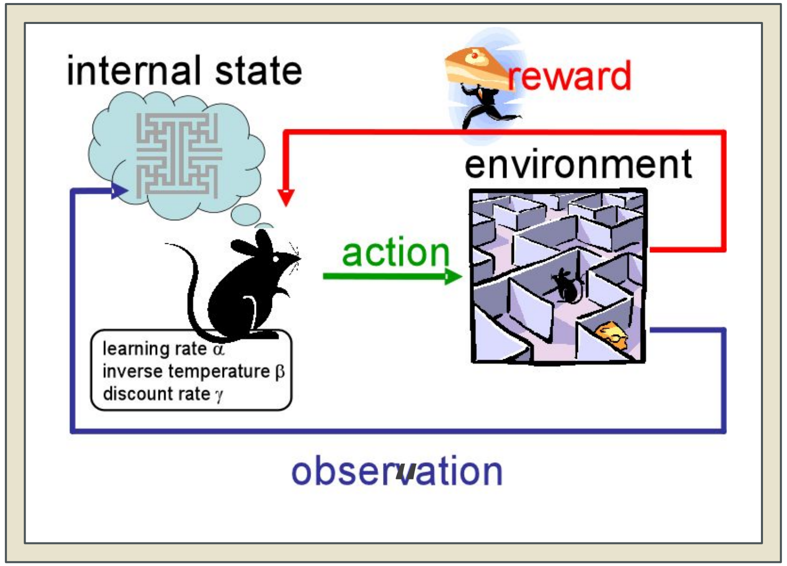
What is Flappy Bird?

TAP the **BIRD** to **FLAP**

Show **evolution** of the Bird as it *'Flaps'* through!

How do we teach the bird?

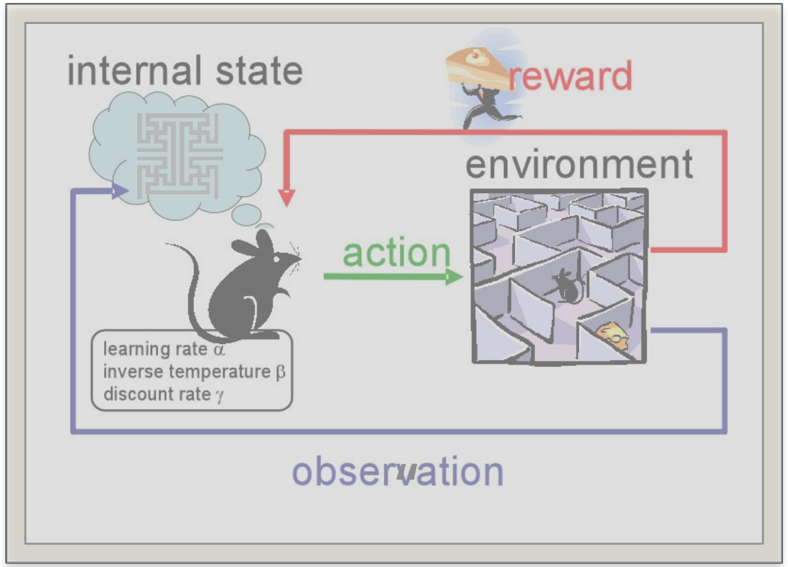
Reinforcement Learning



Make the bird
learn **strategies**

How do we teach the bird?

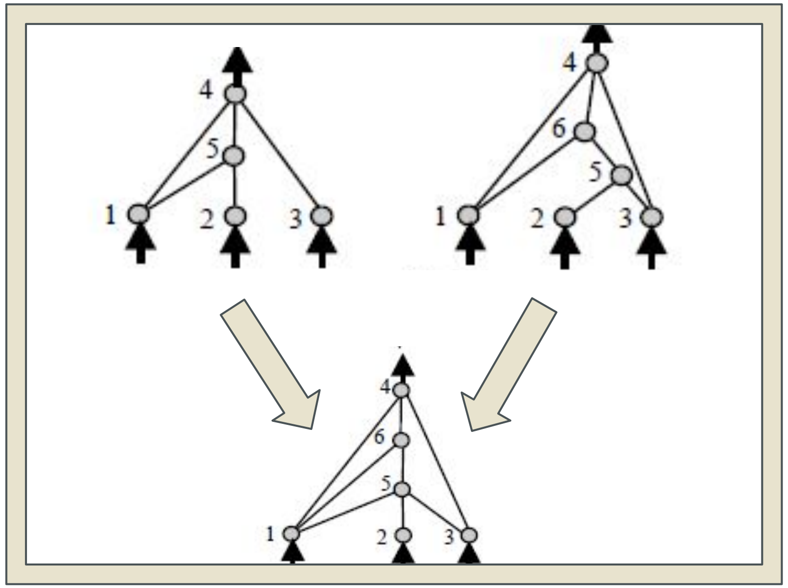
Reinforcement Learning



Make the bird
learn **strategies**

V.S

NEAT Algorithm

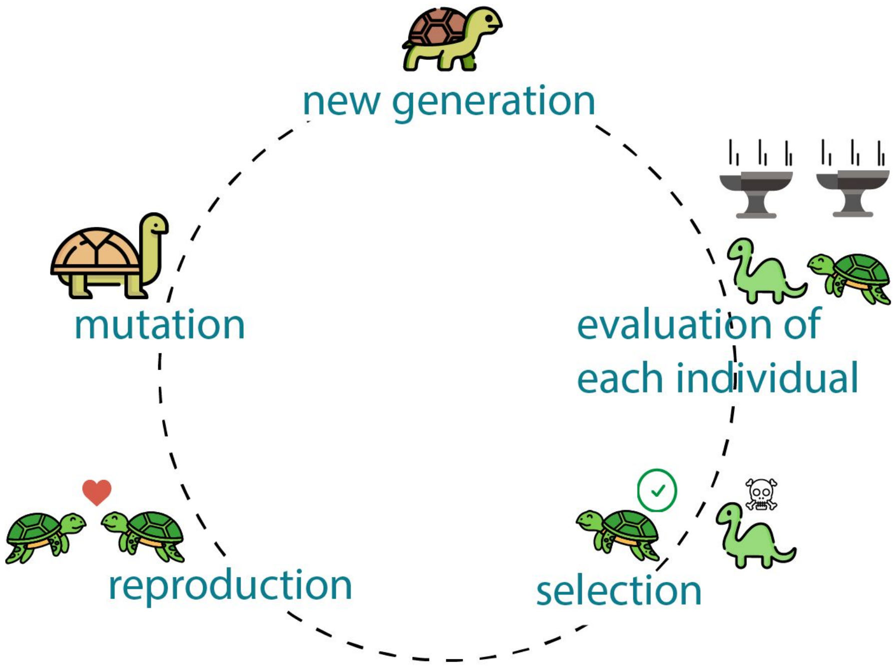


Evolutionarily build the
brain of the bird

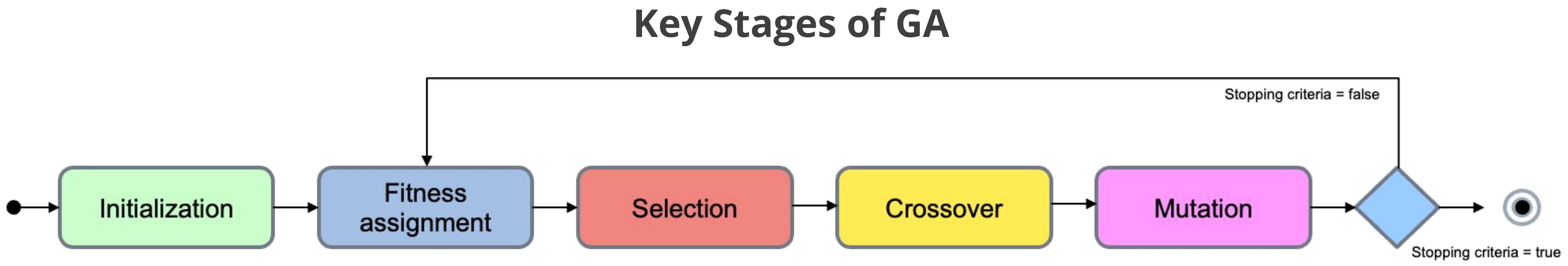
Overview of Algorithms

What is Genetic Algorithm?

Genetics Algorithm (GA) is a search-based technique that find the **best individuals** through the process of **natural selection**

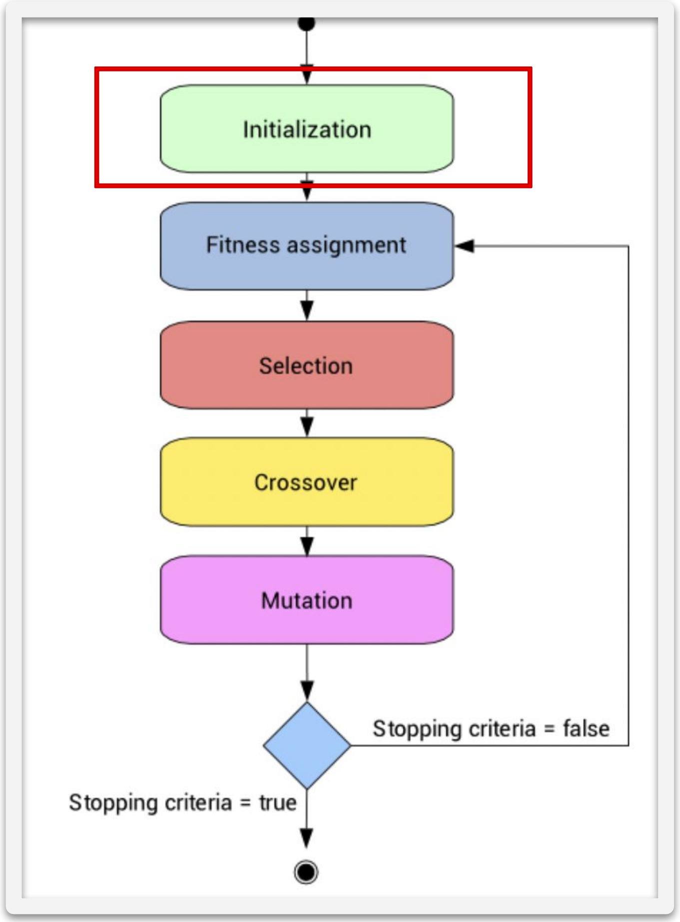


GA Step by Step



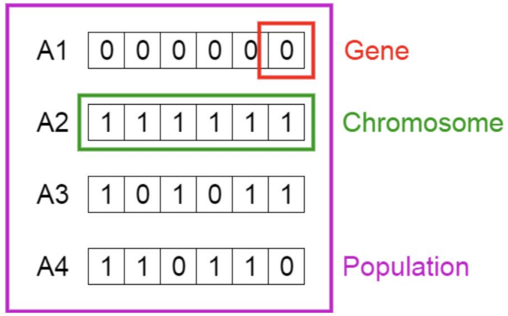
Ref: https://www.analyticsvidhya.com/blog/2017/07/introduction-to-genetic-algorithm/

Step 1: Initialization



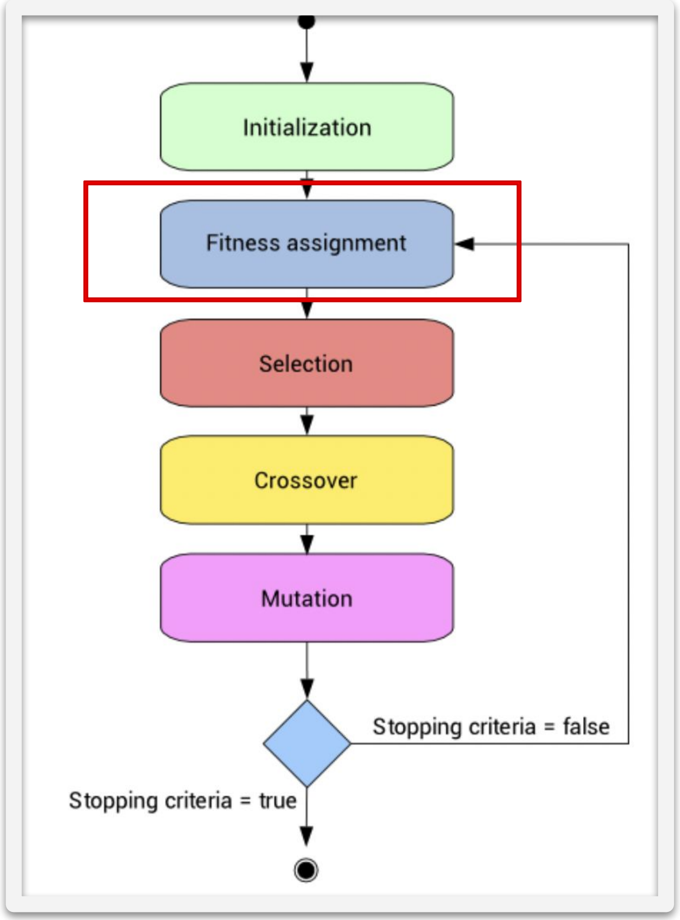
Initial Population

- Each individual is a potential solution
- An individual characterized by a set of **Genes**



Ref: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3

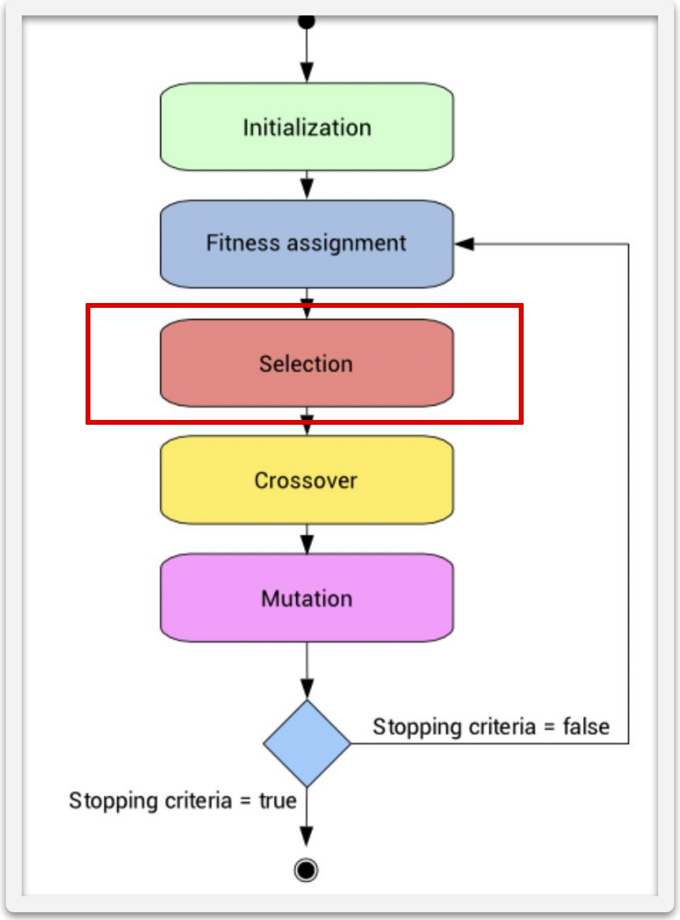
Step 2: Fitness Assignment



Fitness Assignment

- **Fitness score** is calculated for each individual
- Measure **suitability** of an individual to the problem

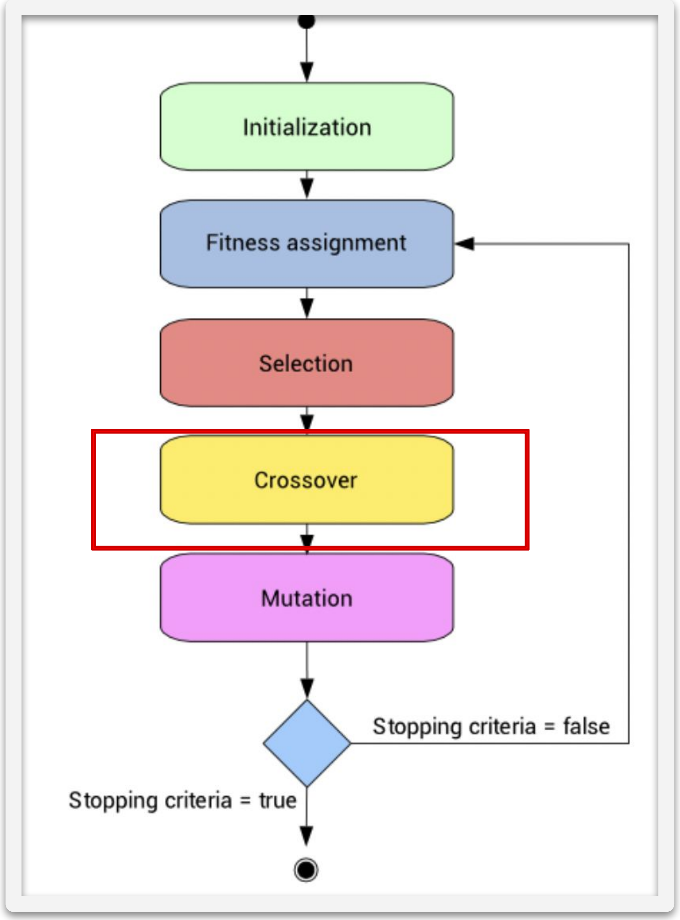
Step 3: Selection



Selection

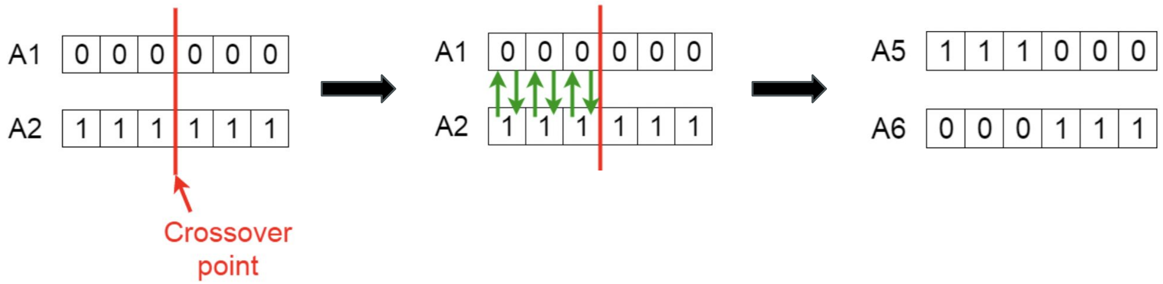
- Measure “**fittestness**” by individual fitness scores
- Fittest individuals are selected to **pass their genes** to next generation

Step 4: Crossover

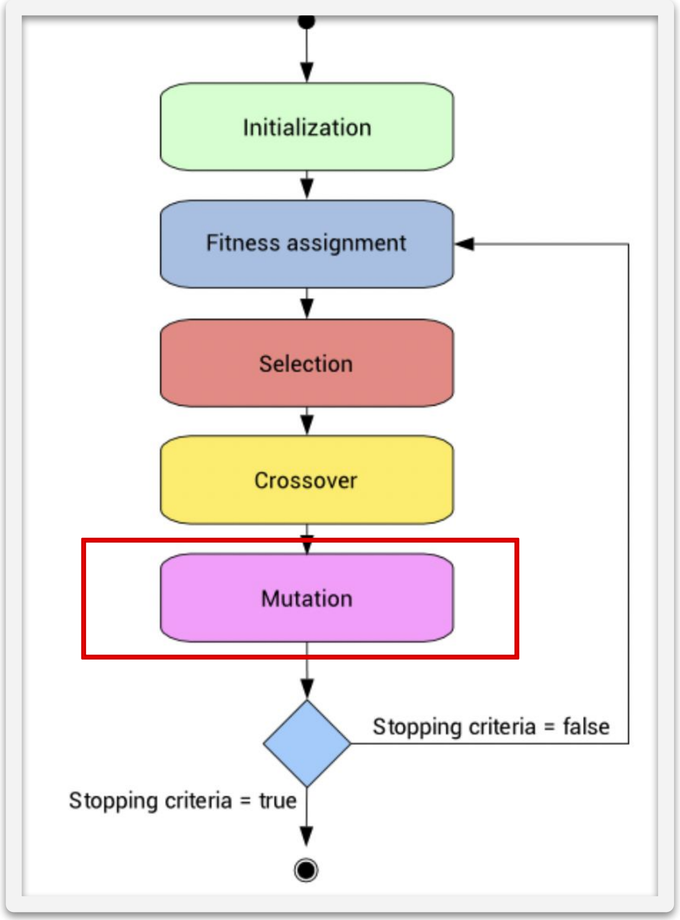


Crossover

- A **crossover point** is chosen randomly
- New offsprings are created and added to the population



Step 5: Mutation



Mutation

- Some offspring genes subjected to **mutation** with a random probability

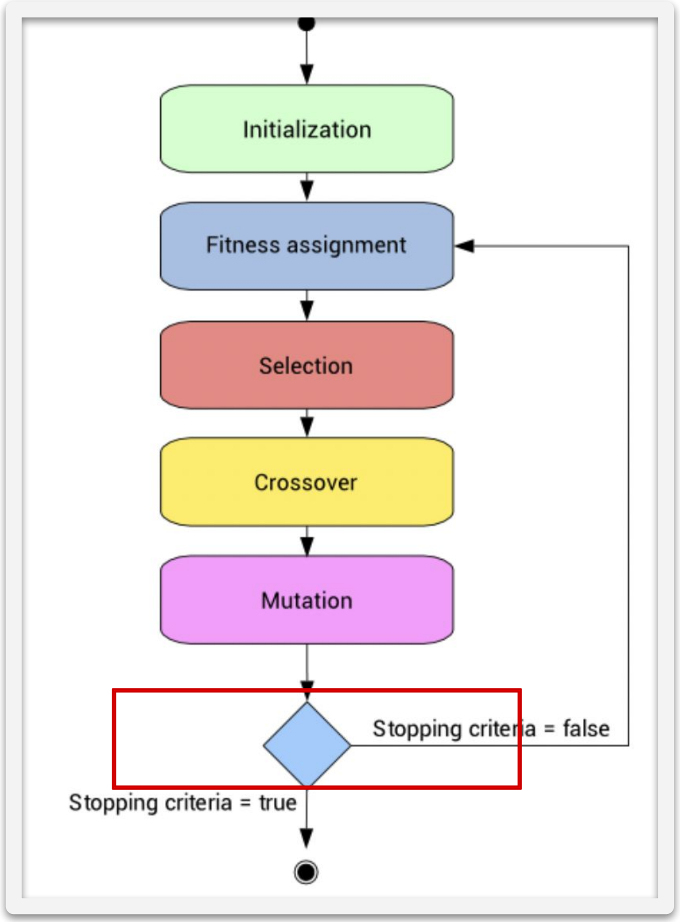
Before Mutation

A5	1	1	1	0	0	0

After Mutation

A5	1	1	0	1	1	0

Step 6: Termination



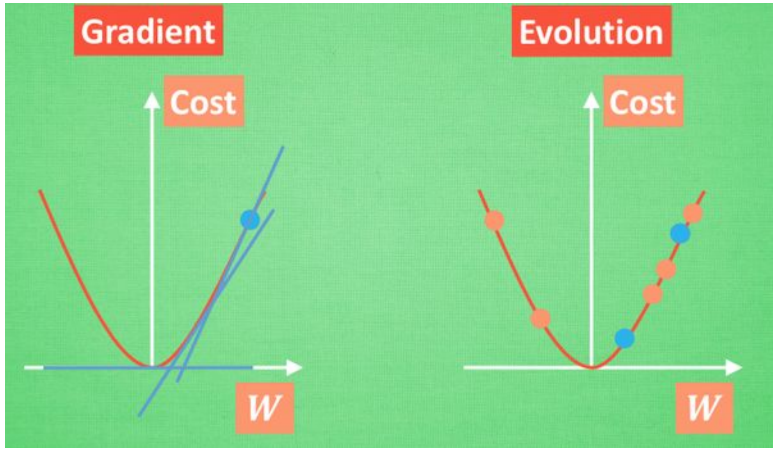
Termination

- When population **converges**
- Some other user-defined stopping criteria

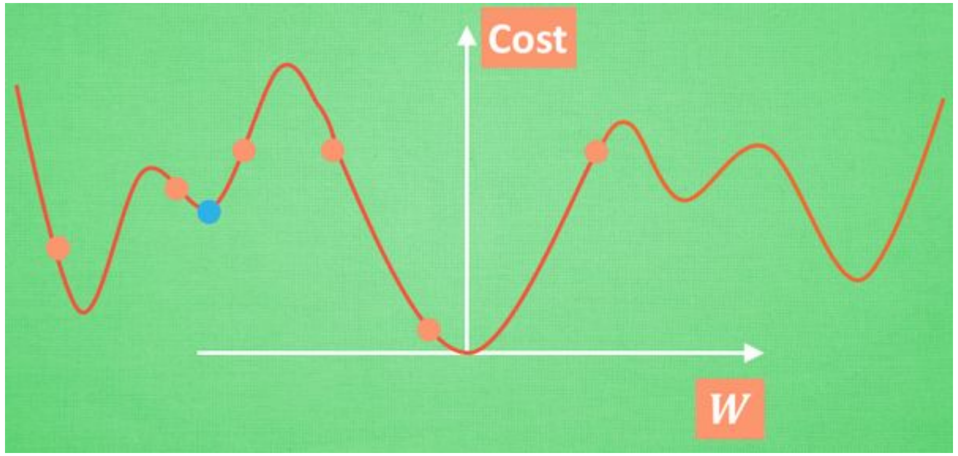
What is NEAT?

NeuroEvolution of Augmenting Topologies (NEAT)

- A type of Genetics Algorithm
- Simultaneously changing parameter weights & network structures



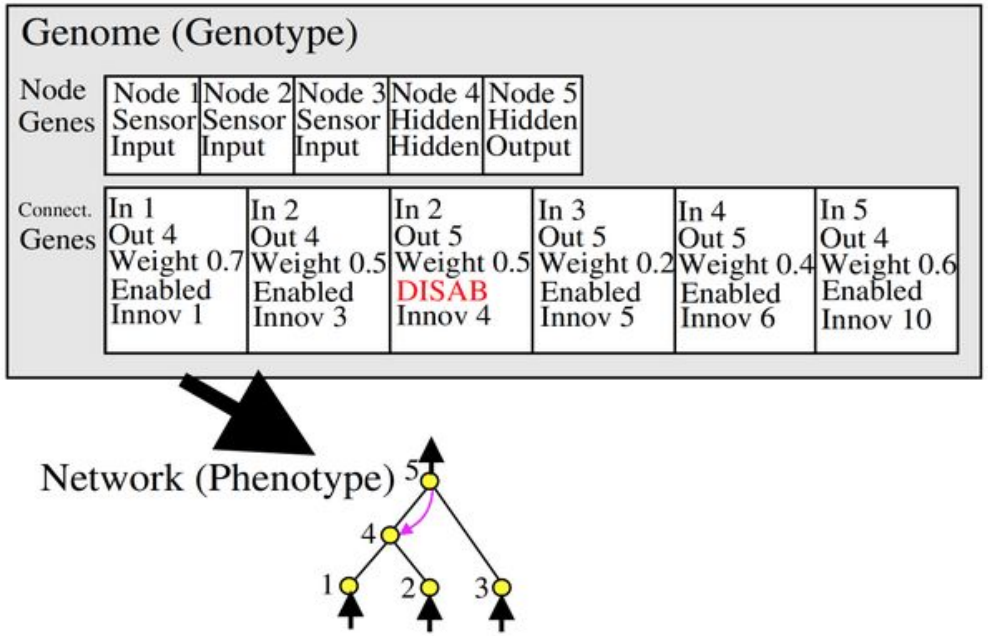
Disadvantage: Slower to Implement



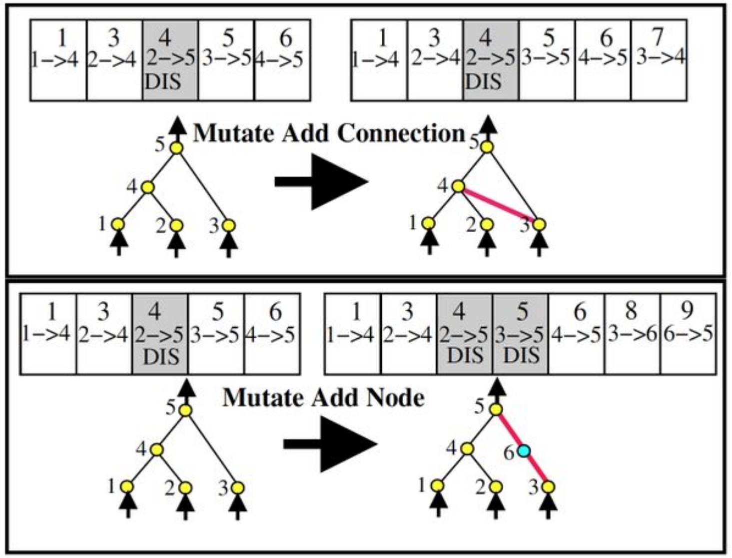
Advantage: More likely to find global solution

How does NEAT work?

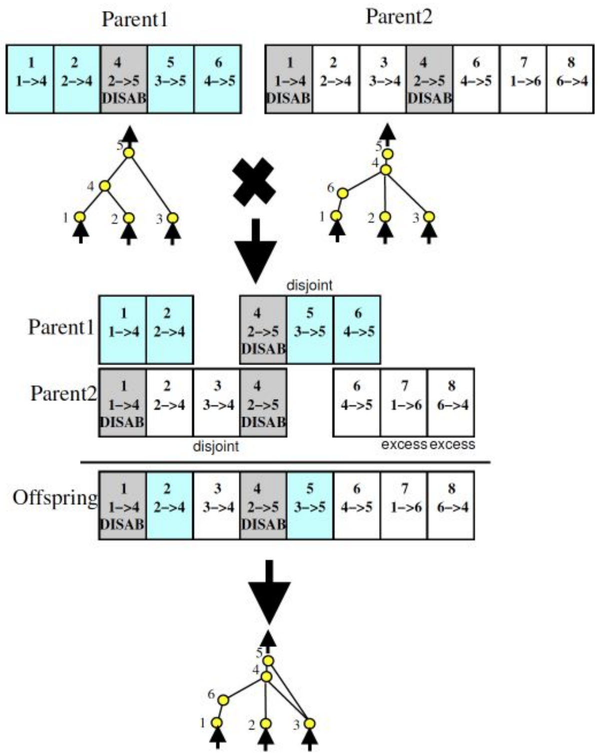
Genetic Encoding



Mutation

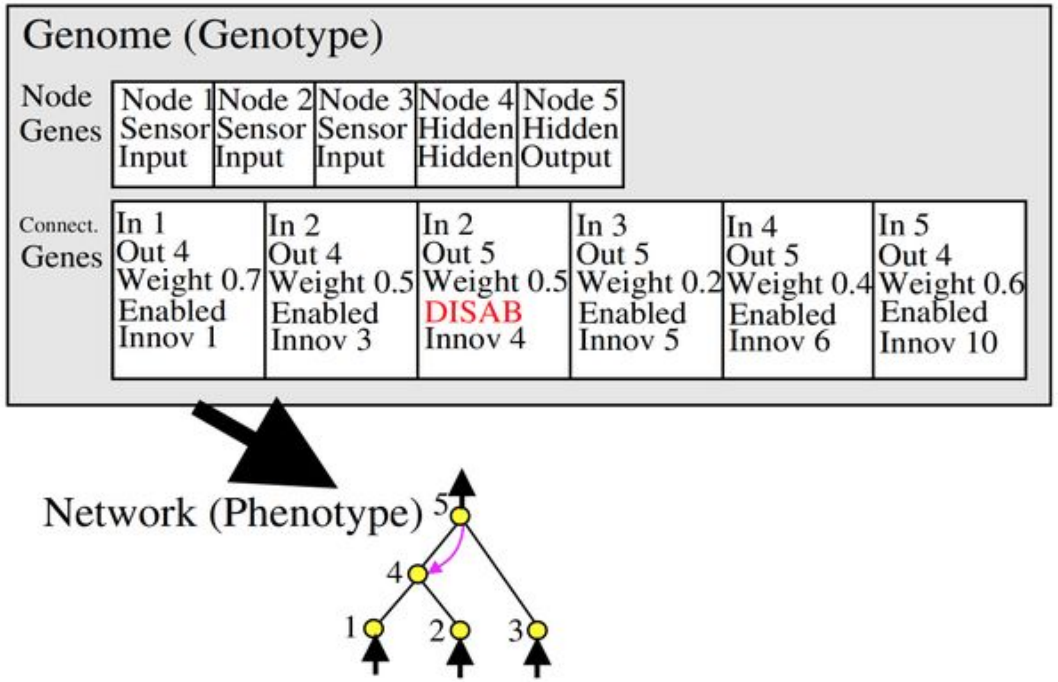


Crossover



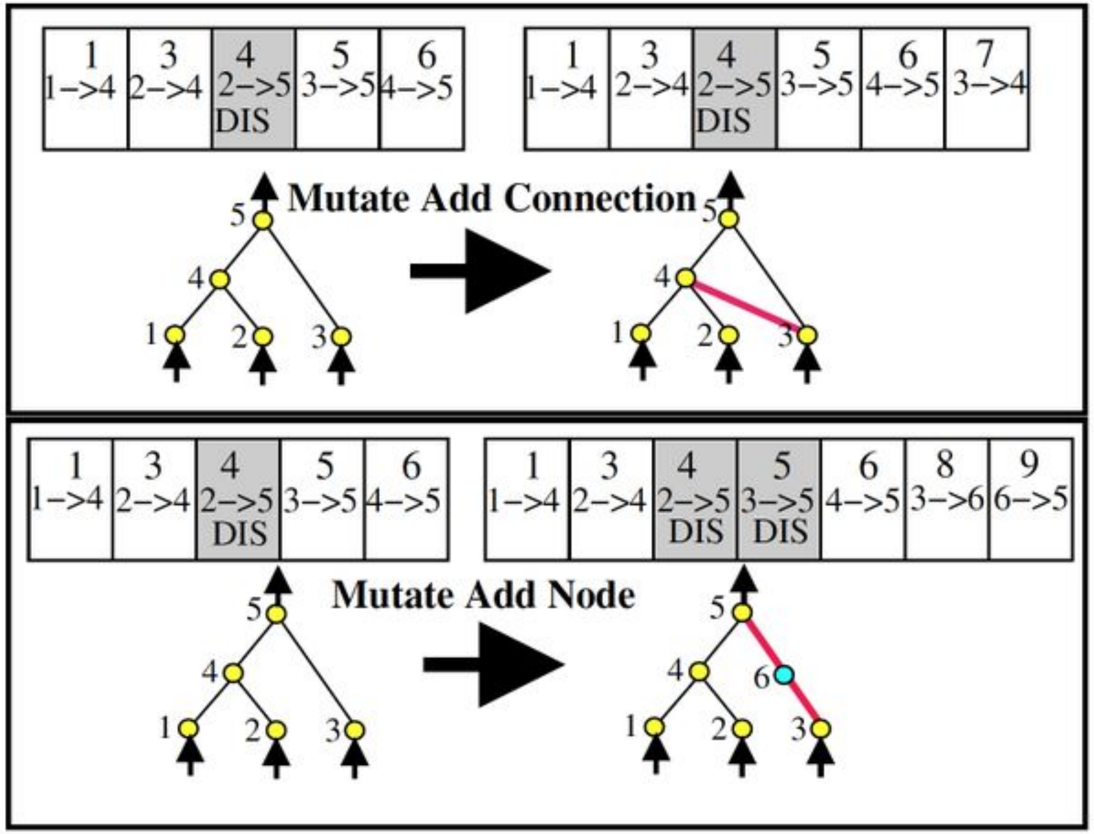
Ref: https://blog.otoro.net/2016/05/07/backprop-neat/

Genetic Encoding



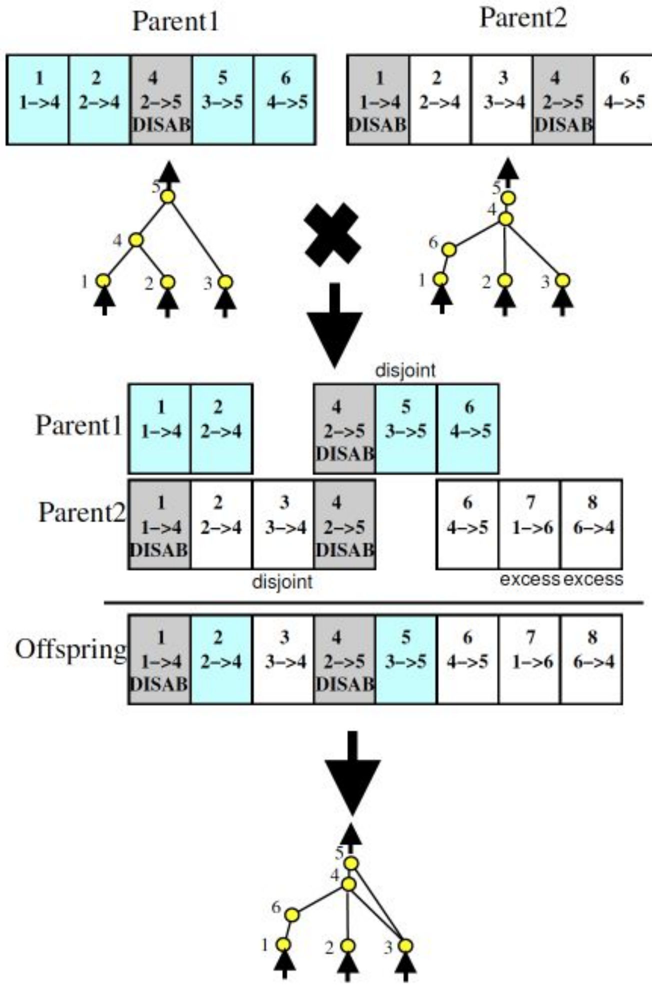
- **Node Genes**
- **Connection Genes**
  - Specifies the *edge* of structure
  - *Offsprings* are list of connection genes

Mutation



- **Type of Mutation**
  - Mutate Connection
  - Mutate Node
- Each mutation action have a unique **innovation number**

Crossover



- **Connections genes** are matched by innovation number

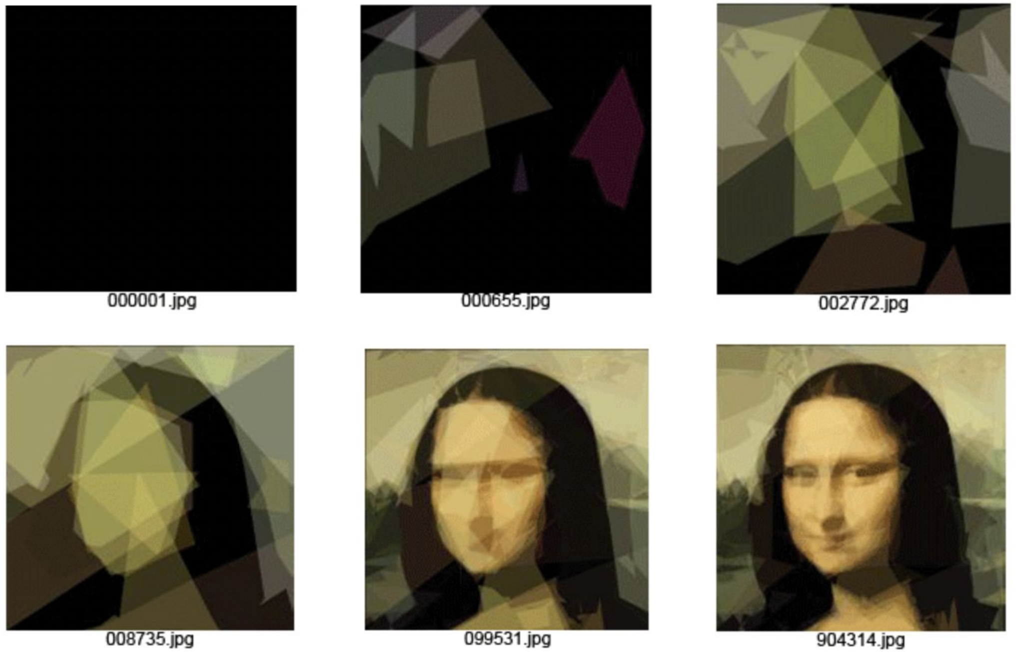
- **Inheritance**:
  a. Both parents contain the same mutation → Randomly inherit one
  b. Only one parent contains the mutation → Inherit directly

Algorithm Applications

How is GA used?

- **Solutions to**:
  - Optimization Problems
  - Multimodal optimization Problems

- **Real-life Applications in**:
  - Economics (e.g. Stock trading)
  - Image Processing (e.g. Mona Lisa)

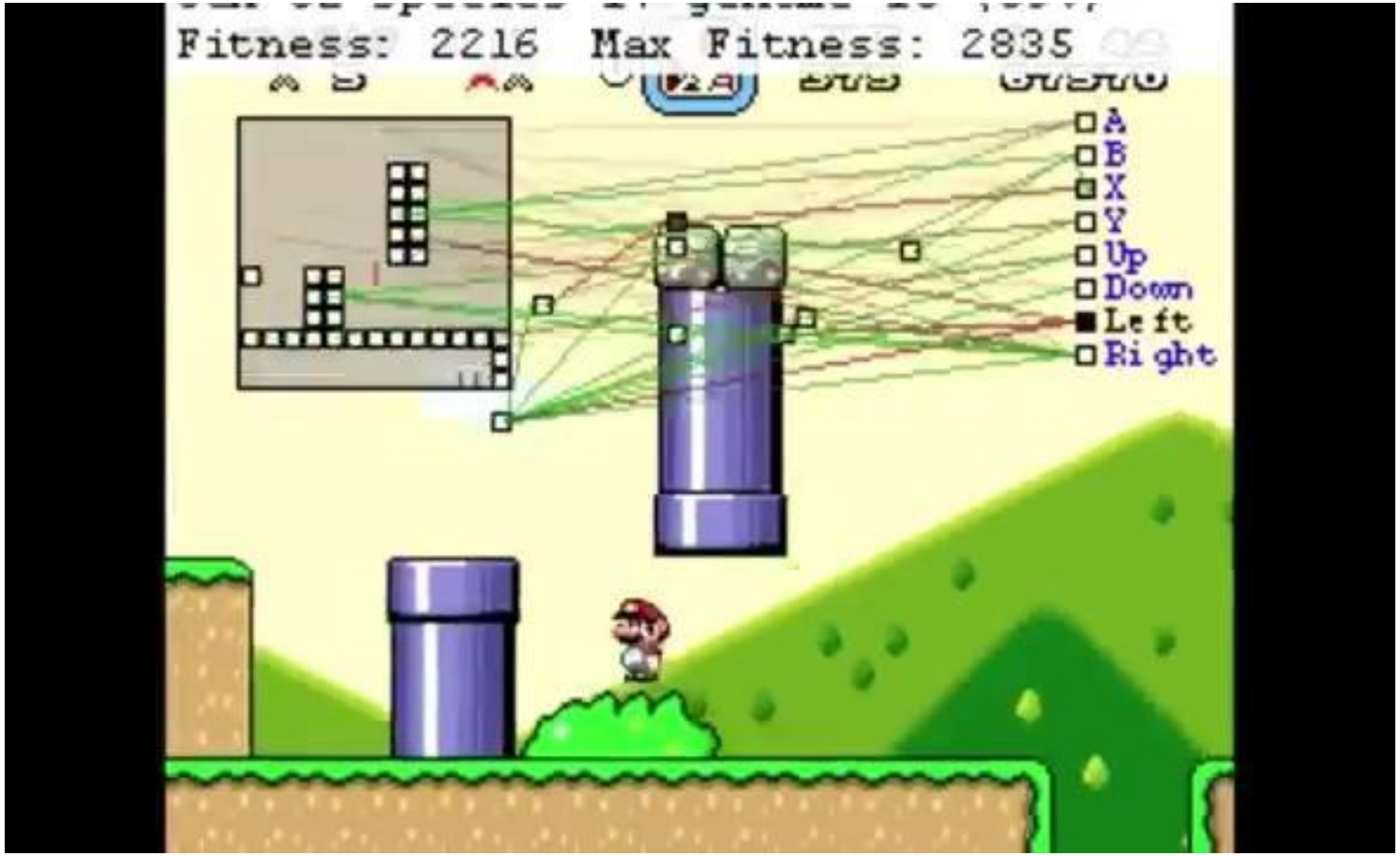


Ref: https://rogerjohansson.blog/2008/12/07/genetic-programming-evolution-of-mona-lisa/

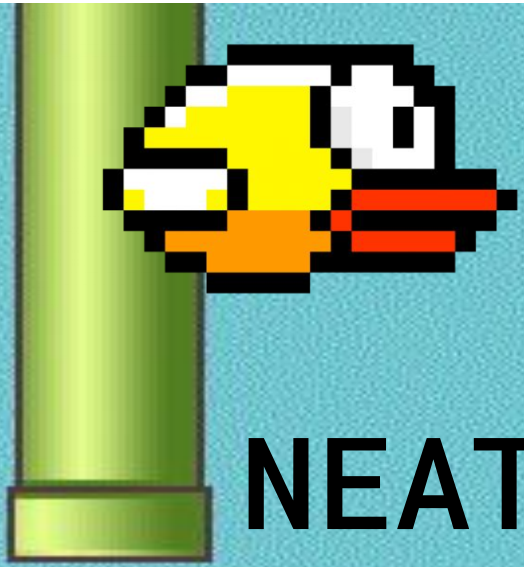
How is NEAT used?

Applications in:

- General game playing
- Self-driving cars
- Simulation of microorganism evolution

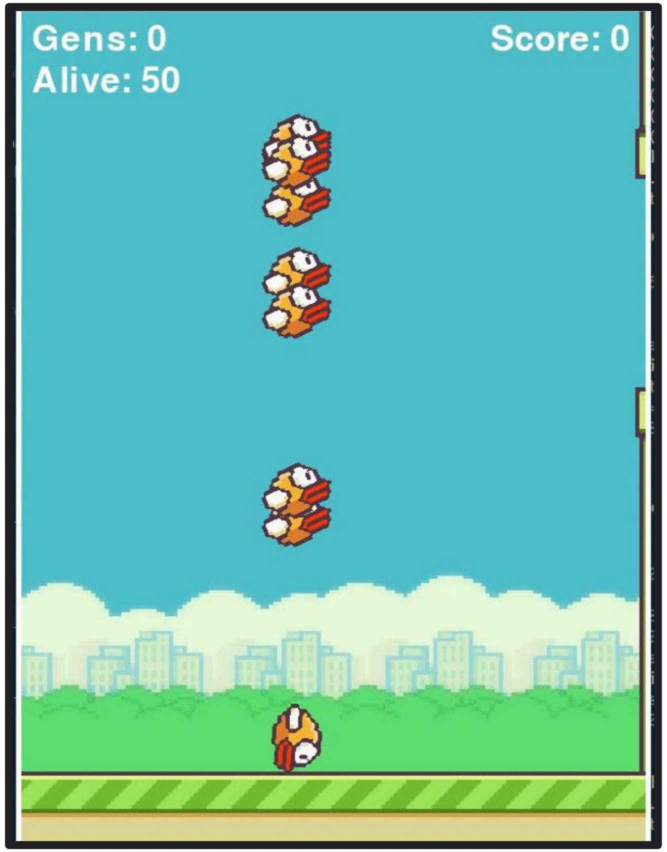


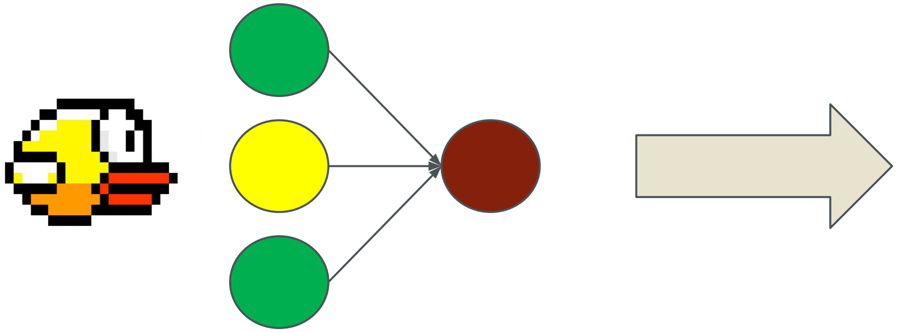
Ref: https://mofanpy.com/static/results/evolutionary-algorithm/4-1-0.mp4



NEAT in Action

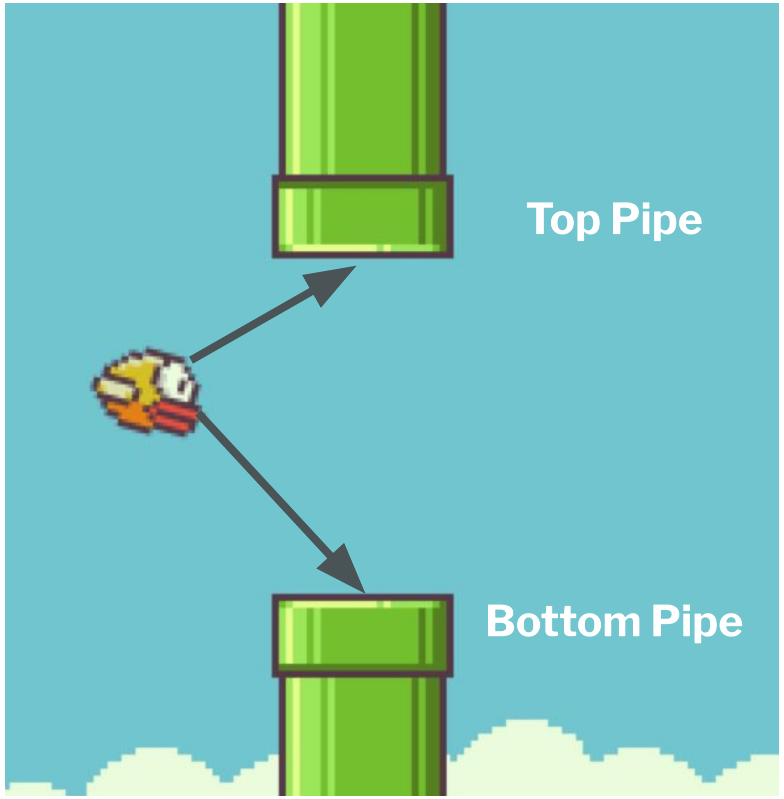
Training the Bird





FITNESS SCORE=
[0.1 * Num Frames]

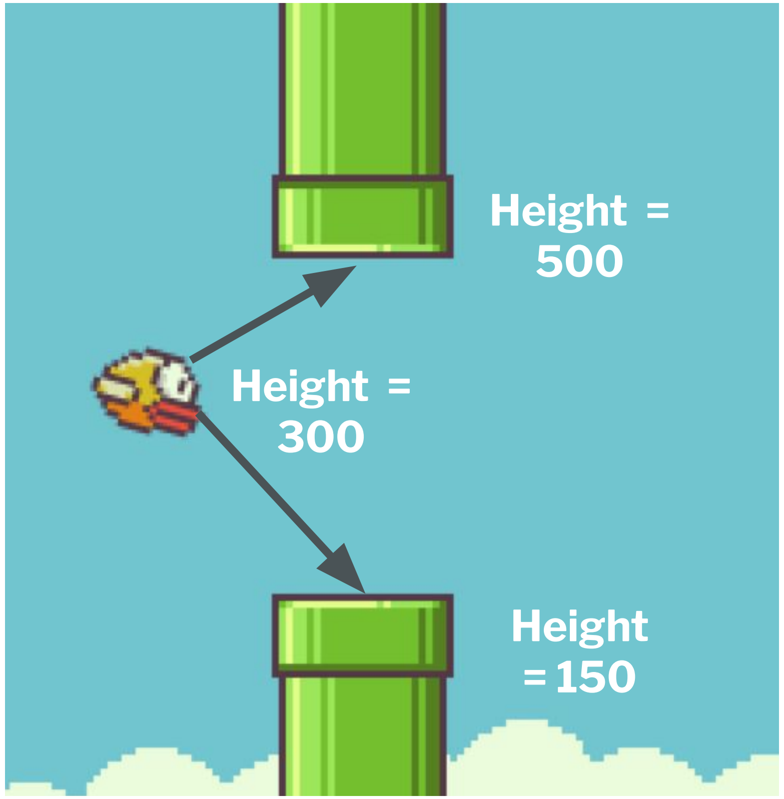
Behind the Scenes



Initial Input Neurons

1. Location of the **Bird**
2. Distance from **Top Pipe**
3. Distance from **Bottom Pipe**

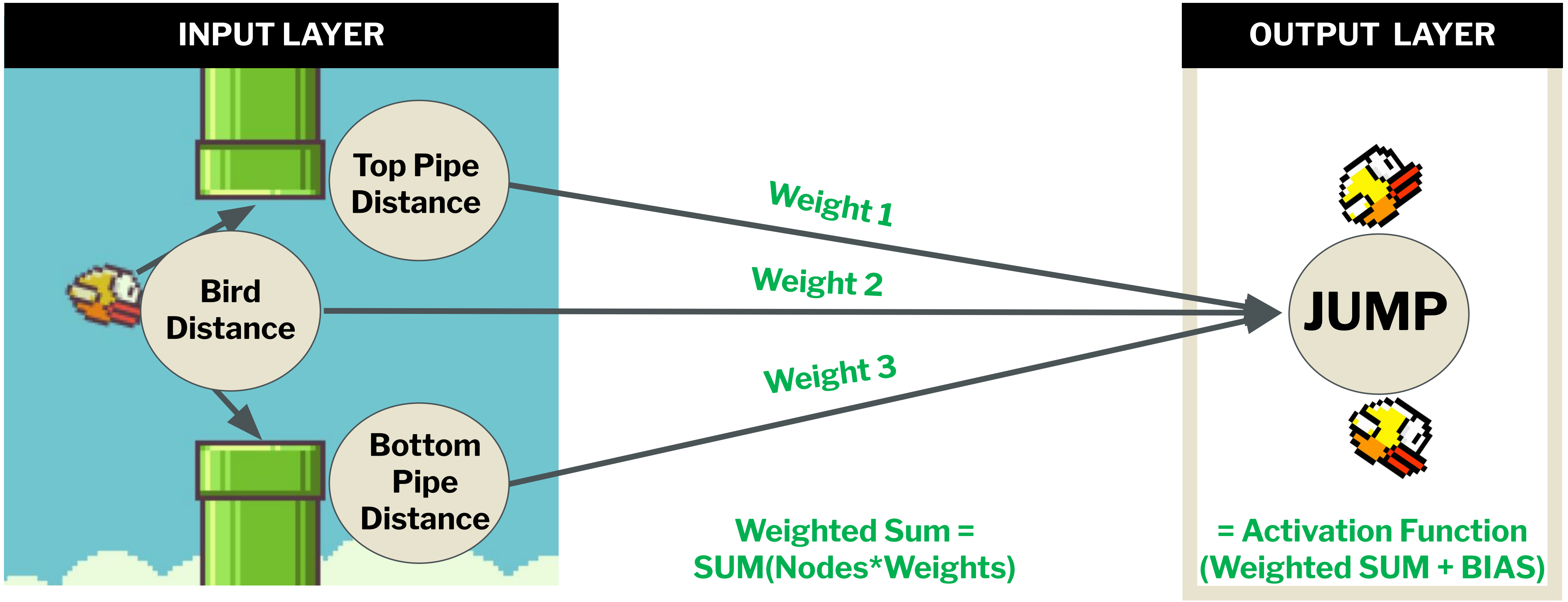
Behind the Scenes



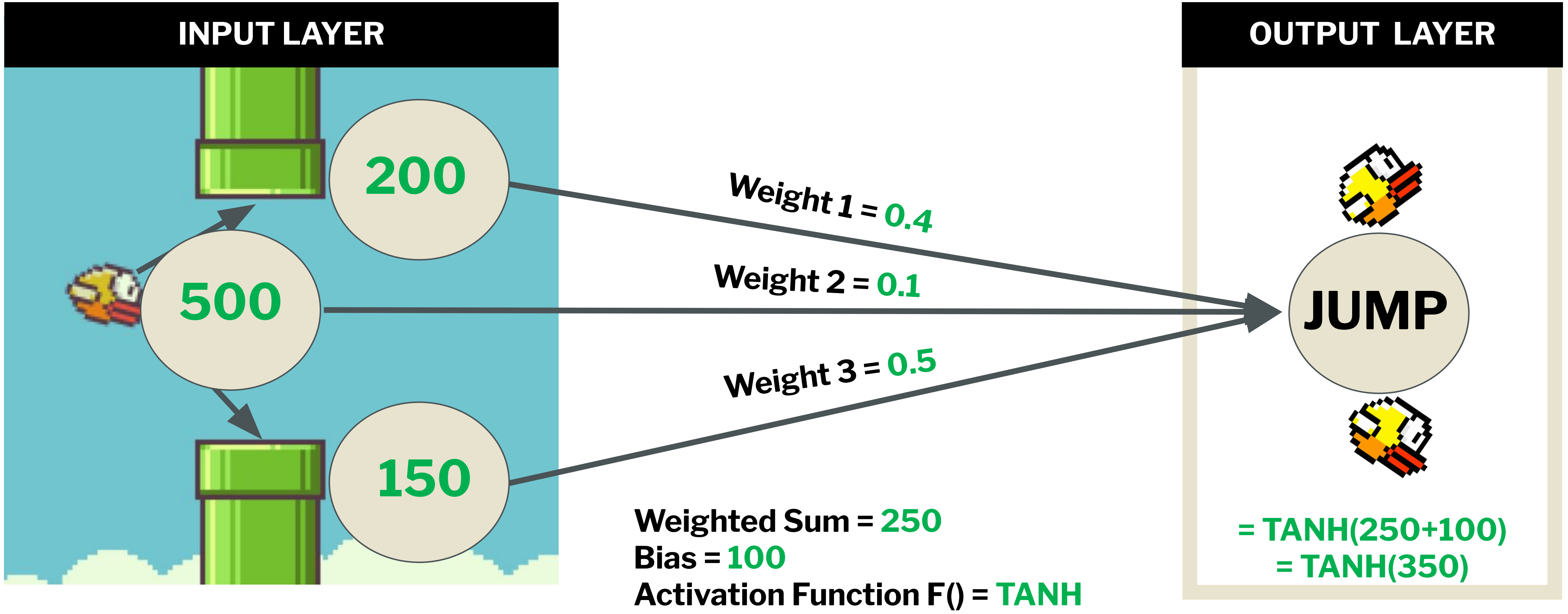
Initial Input Neurons

1. Location of the Bird = **300**
2. Distance from Top Pipe = **200** (500-300)
3. Distance from Bottom Pipe = **150** (300-150)

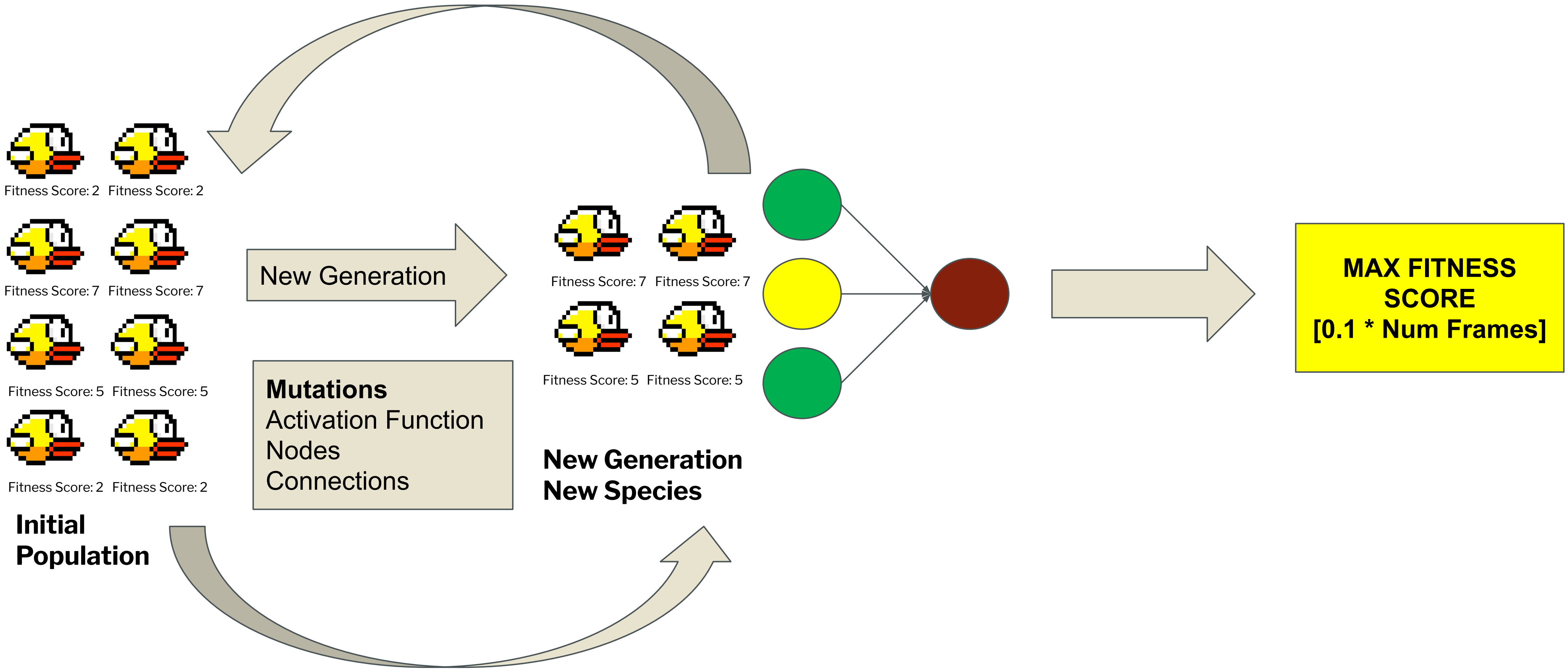
Initial Neural Network



Back to the example:



Evolution of the Network



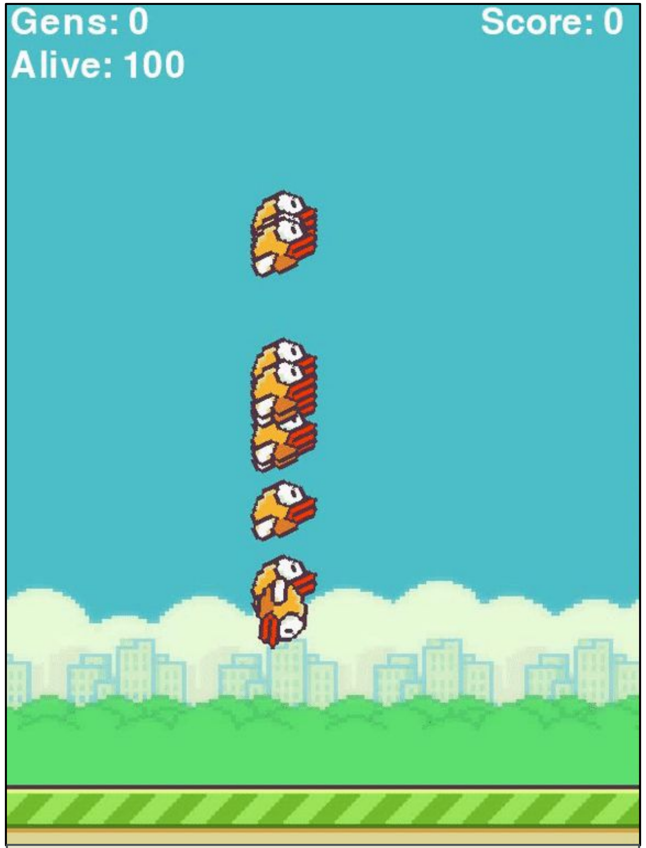
Key Hyperparameters

Hyperparameters	***Influence***
Population Size	Influences starting fitness score
Node Add/Remove Probability	Adjusts topology of neural network
Mutation Rate	Influences probability of species changing
Number of Hidden Layer Nodes	Initial number of nodes in hidden layer
Maximum Stagnation	Removes species that have not improved after n generations
Change Activation Function*	Activation functions for each layer
Compatibility Threshold	Lower the threshold the more species in a population

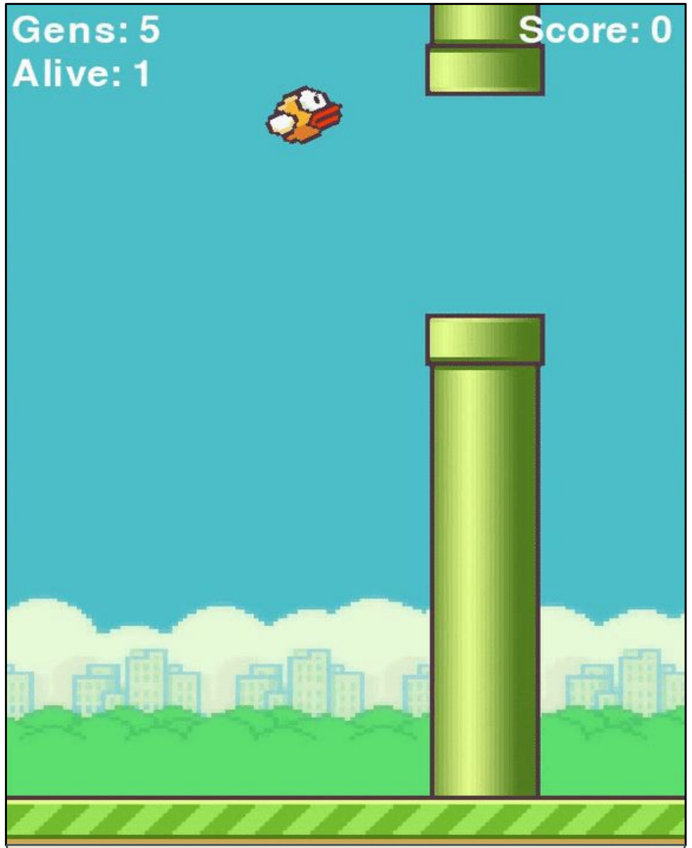
Key Hyperparameters

Hyperparameters	***Influence***
Population Size	**Influences starting fitness score**
Node Add/Remove Probability	Adjusts topology of neural network
Mutation Rate	**Influences probability of species changing**
Number of Hidden Layer Nodes	Initial number of nodes in hidden layer
Maximum Stagnation	**Removes species that have not improved after n generations**
Change Activation Function*	Activation functions for each layer
Compatibility Threshold	**Lower the threshold the more species in a population**

Population Size

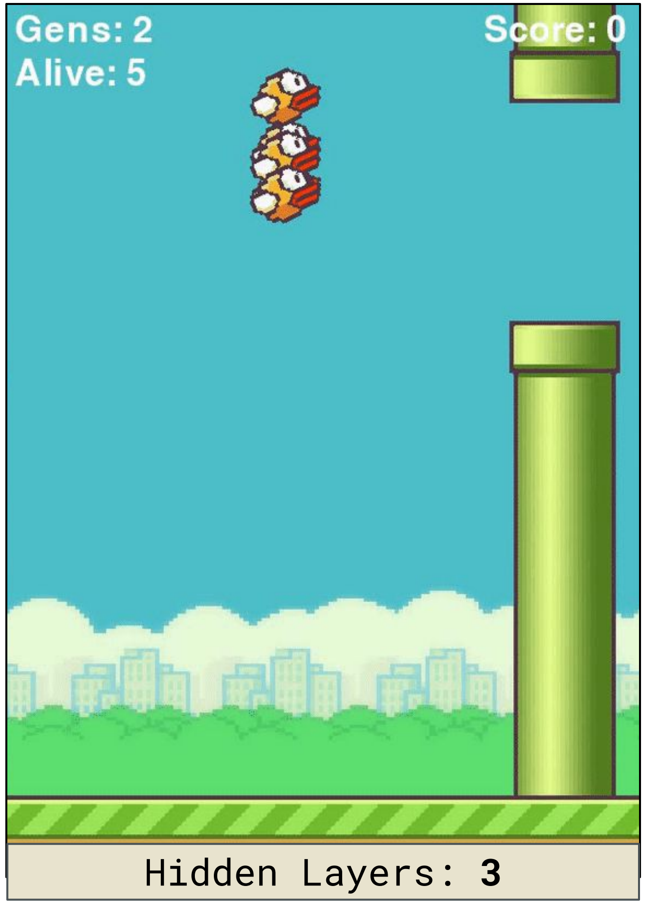


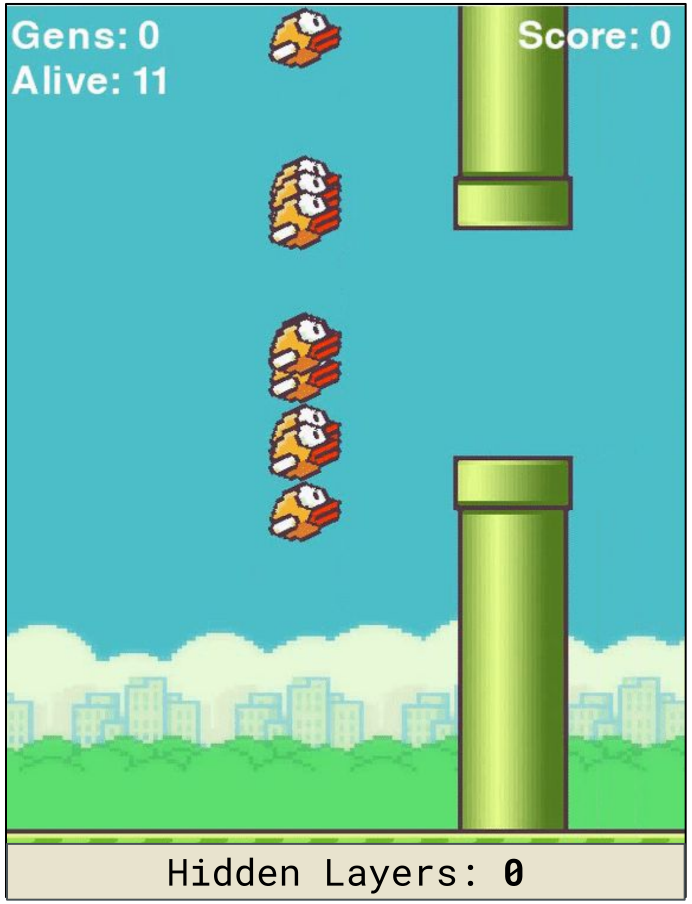
Population Size: **100**



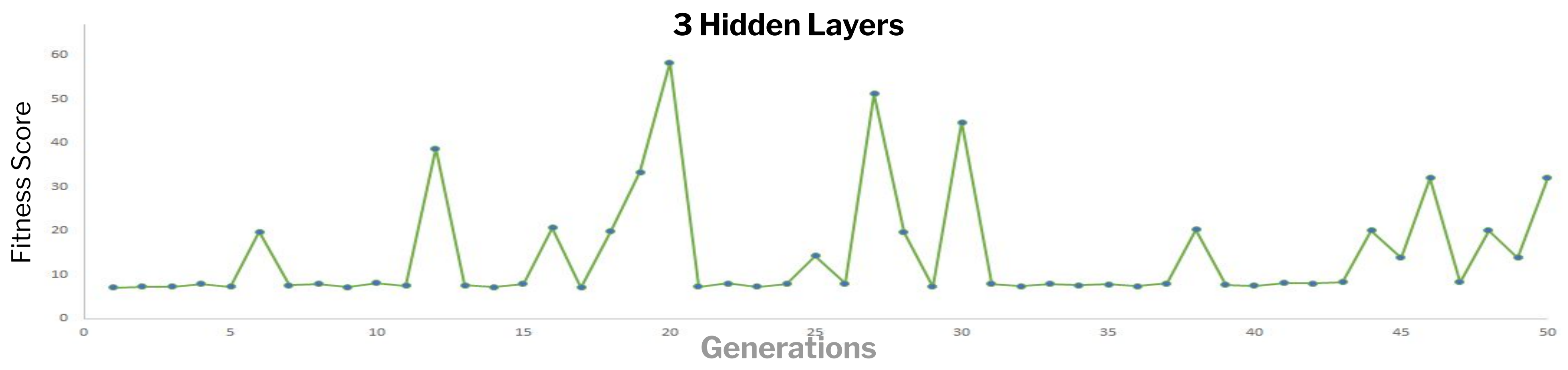
Population Size: **10**

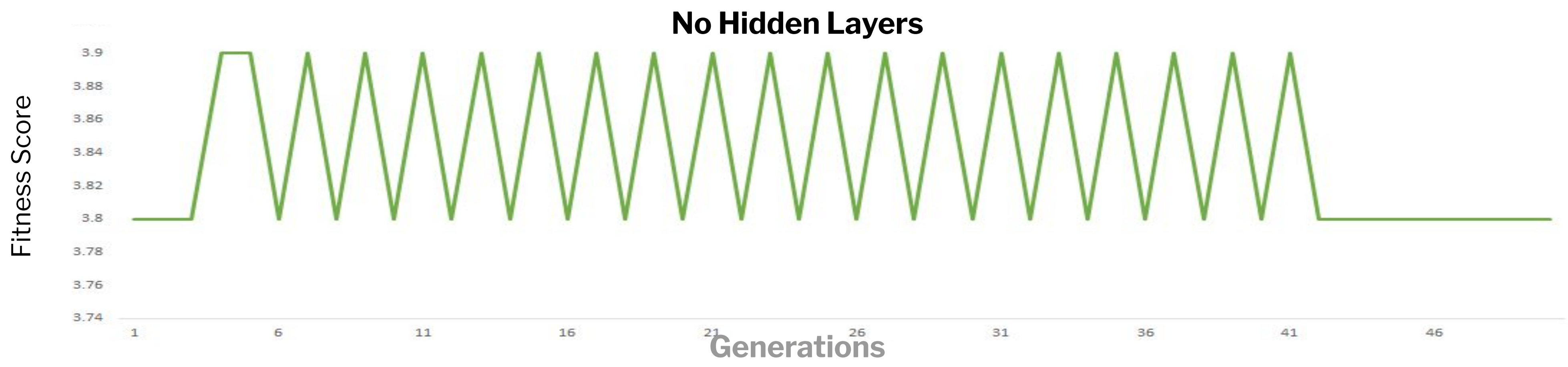
Hidden Layers





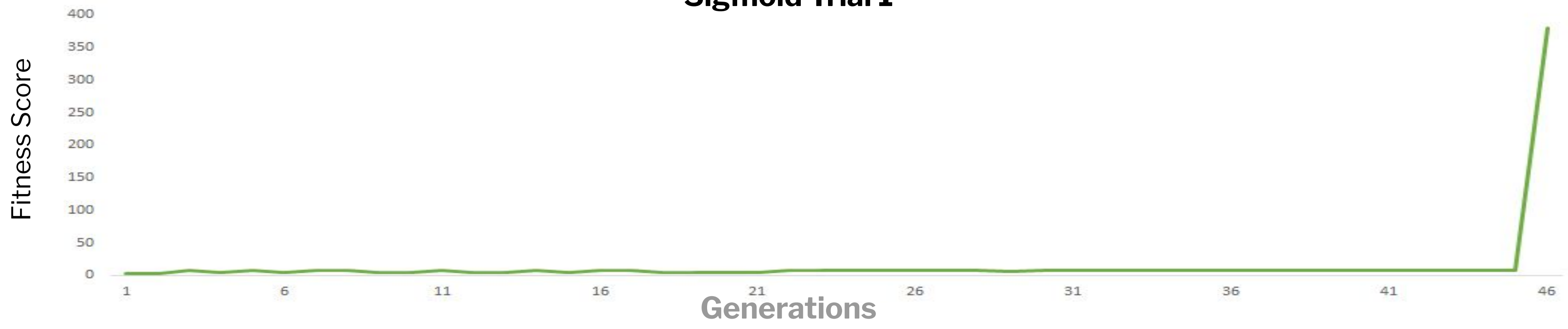
Hidden Layers



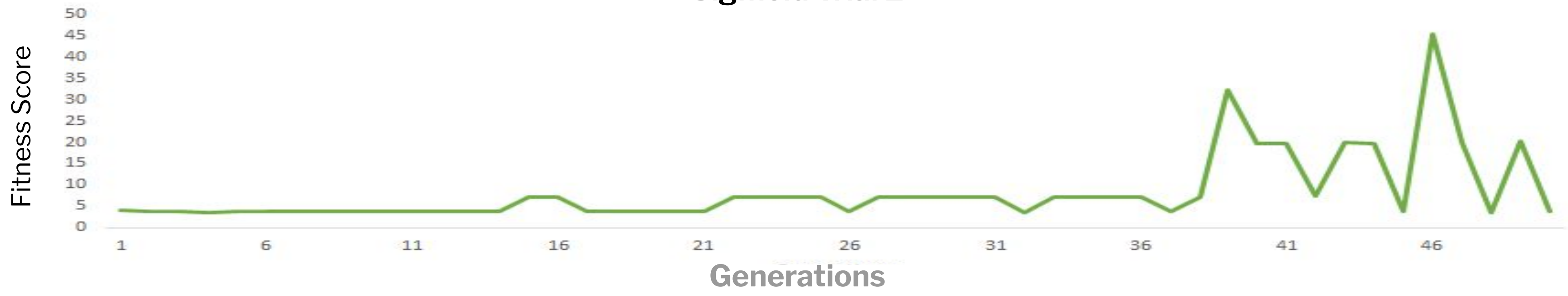


Comparing Model Runs

Sigmoid Trial 1



Sigmoid Trial 2



Limitations of Model

1. **Population Size** is by far the most important parameter, but requires high compute power
2. **Randomness or seed** can influence performance
3. **Mutation** can be disruptive lead to poorer performance in the short term
4. **Exploratory** instead of iterative

Before Mutation

A5	1	1	1	0	0	0

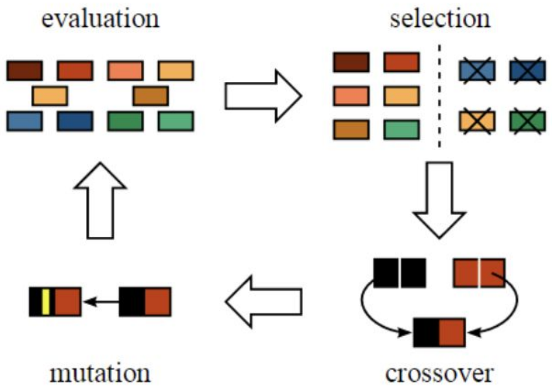
After Mutation

A5	1	1	0	1	1	0

Key Takeaways

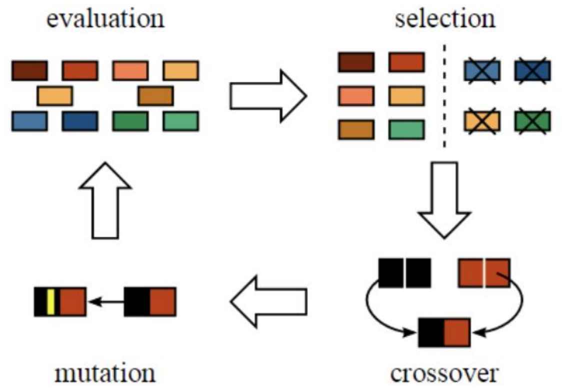
You learned...

- **Genetic Algorithm**
  - Evolutionarily builds neural networks
  - Selection-based that takes the ‘best genes’ from previous to output an offspring

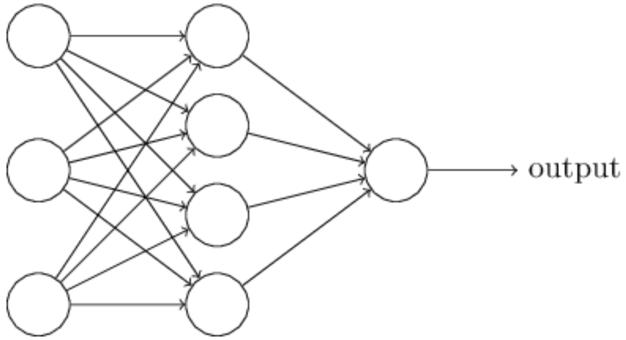


You learned...

- **Genetic Algorithm**
  - Evolutionarily builds neural networks
  - Selection-based that takes the 'best genes' from previous to output an offspring



- **NEAT Algorithm**
  - NeuroEvolution of Augmenting Topologies
  - Simultaneous changes in parameter weight & network structure



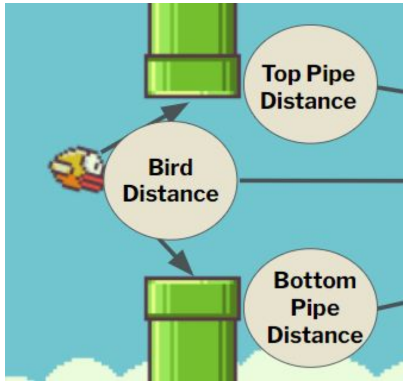
You also saw...

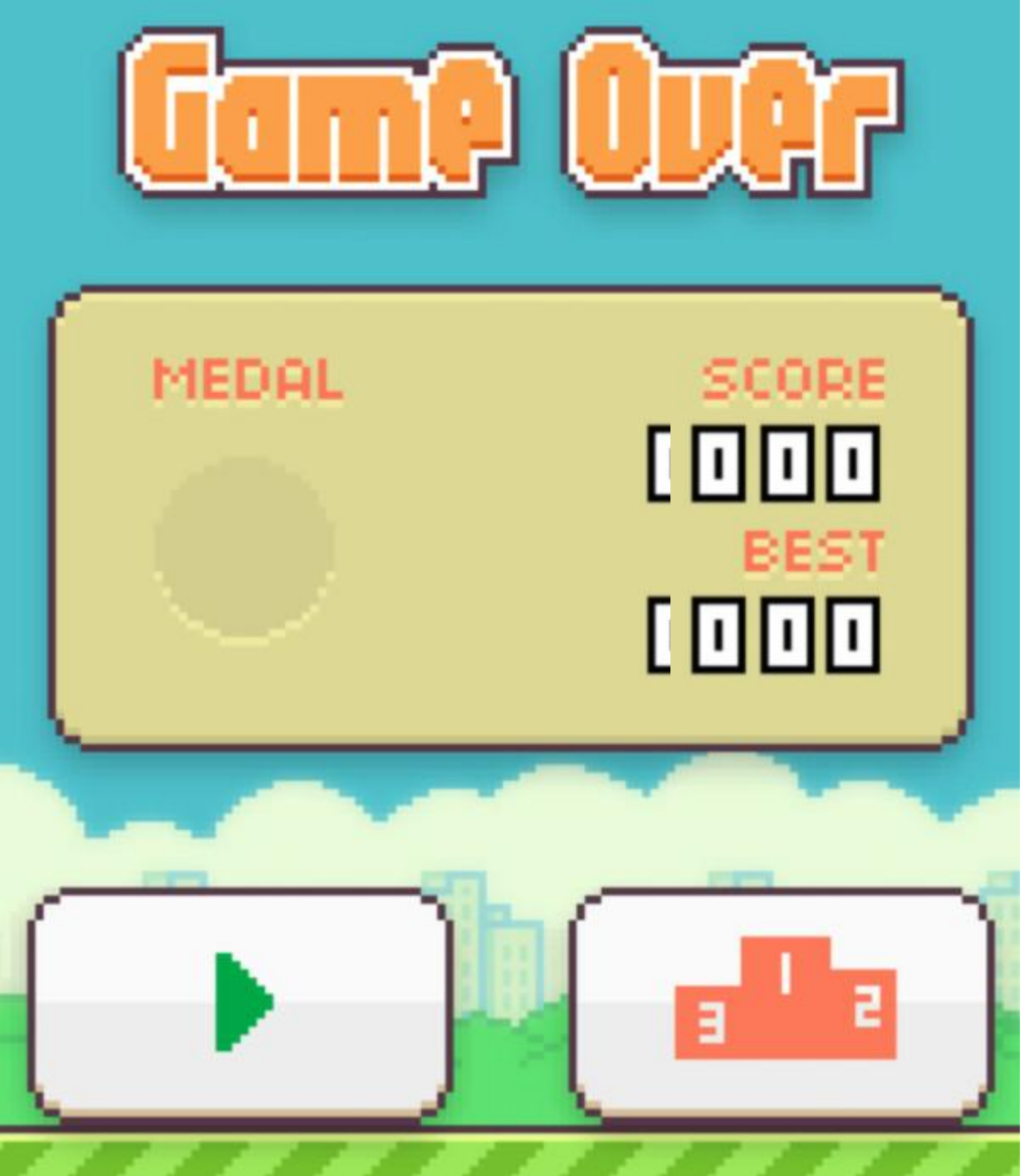
- **Real-world Applications**
  - GA in Optimization problems, Economics, Image processing
  - NEAT in autonomous cars, games, microorganism evolution

You also saw...

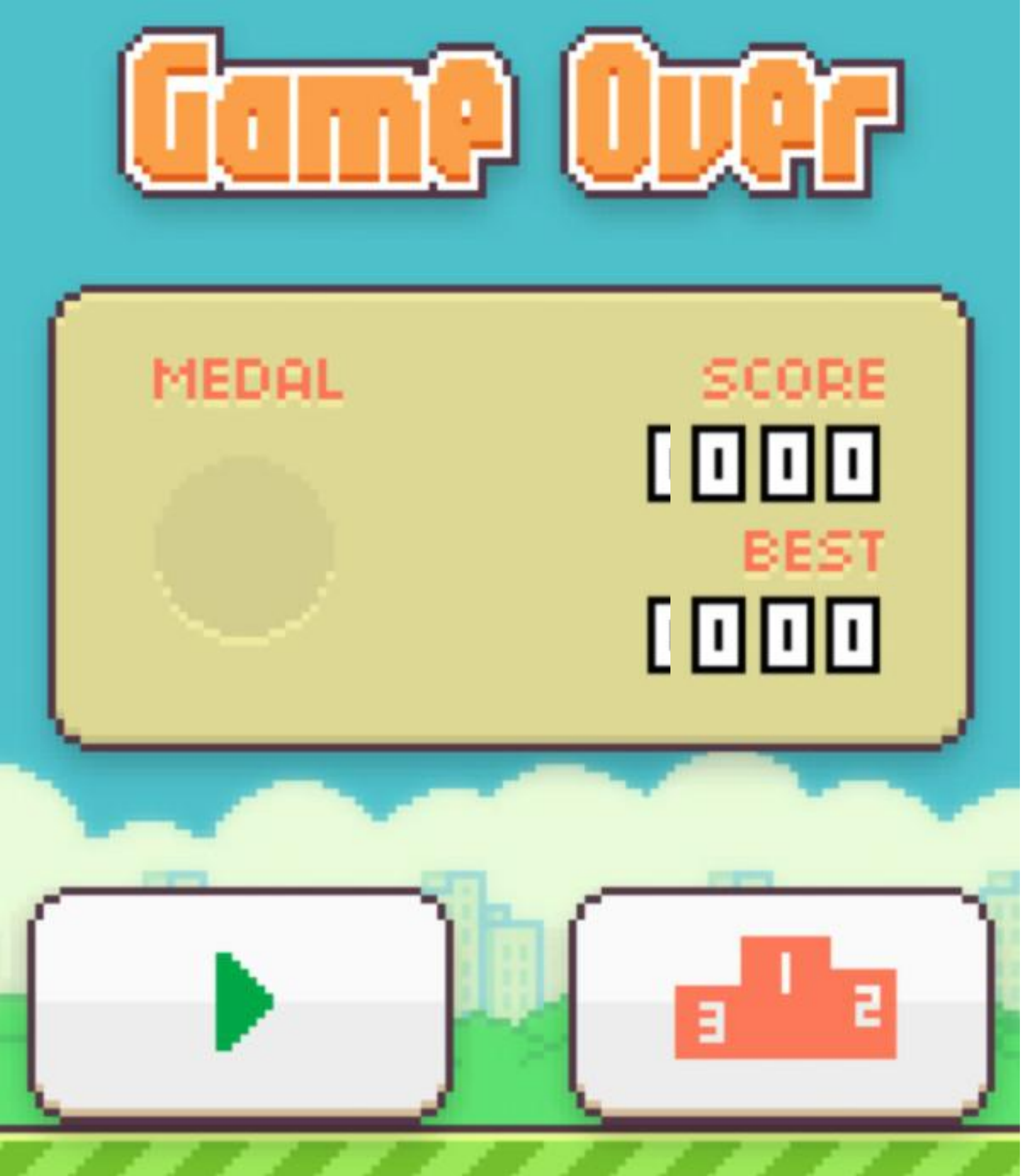
- **Real-world Applications**
  - GA in Optimization problems, Economics, Image processing
  - NEAT in autonomous cars, games, microorganism evolution

- **Flappy Bird**
  - Start with key initial nodes, evolutionarily build the network
  - Effects of Hyperparameter Tuning





Best Takeaway:
how to get the
best score for
Flappy Bird ;)



Thank you!

References

Stanley, K., & Miikkulainen, R. (n.d.). Efficient evolution of neural network topologies. *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*. doi:10.1109/cec.2002.1004508

JAIN, S. (2017). Genetic Algorithm | Application Of Genetic Algorithm. Retrieved 2 May 2021, from https://www.analyticsvidhya.com/blog/2017/07/introduction-to-genetic-algorithm/

Mallawaarachchi, V. (2017). Introduction to Genetic Algorithms — Including Example Code. Retrieved 2 May 2021, from https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3

Genetic Algorithms - Application Areas - Tutorialspoint. (n.d.). Retrieved 2 May 2021, from https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_application_areas.htm

Johansson, V. (2008). Genetic Programming: Evolution of Mona Lisa. Retrieved 2 May 2021, from https://rogerjohansson.blog/2008/12/07/genetic-programming-evolution-of-mona-lisa/?fbclid=IwAR1IvWSOCzhICDQHRveCQQR2UTHUUW6UEp3vnM67VA88P6Iso_ZhDIkP1VU

Techwithtim/NEAT-Flappy-Bird. (2020). Retrieved 2 May 2021, from https://github.com/techwithtim/NEAT-Flappy-Bird

Code Output

```
****** Running generation 0 ******

Population's average fitness: 3.81000 stdev: 1.82617
Best fitness: 7.30000 - size: (1, 3) - species 1 - id 10
Average adjusted fitness: 0.288
Mean genetic distance 1.945, standard deviation 0.357
Population of 10 members in 1 species:
   ID   age  size  fitness  adj fit  stag
  ====  ===  ====  =======  =======  ====
     1    0    10      7.3    0.288     0
Total extinctions: 0
Generation time: 2.794 sec
```

```
****** Running generation 1 ******

Population's average fitness: 5.02000 stdev:
2.19718
Best fitness: 7.30000 - size: (2, 4) - species 1 - id
12
Average adjusted fitness: 0.535
Mean genetic distance 1.416, standard deviation
0.410
Population of 10 members in 1 species:
   ID   age  size  fitness  adj fit  stag
  ====  ===  ====  =======  =======  ====
     1    1    10      7.3    0.535     1
Total extinctions: 0
Generation time: 2.786 sec (2.790 average)
```