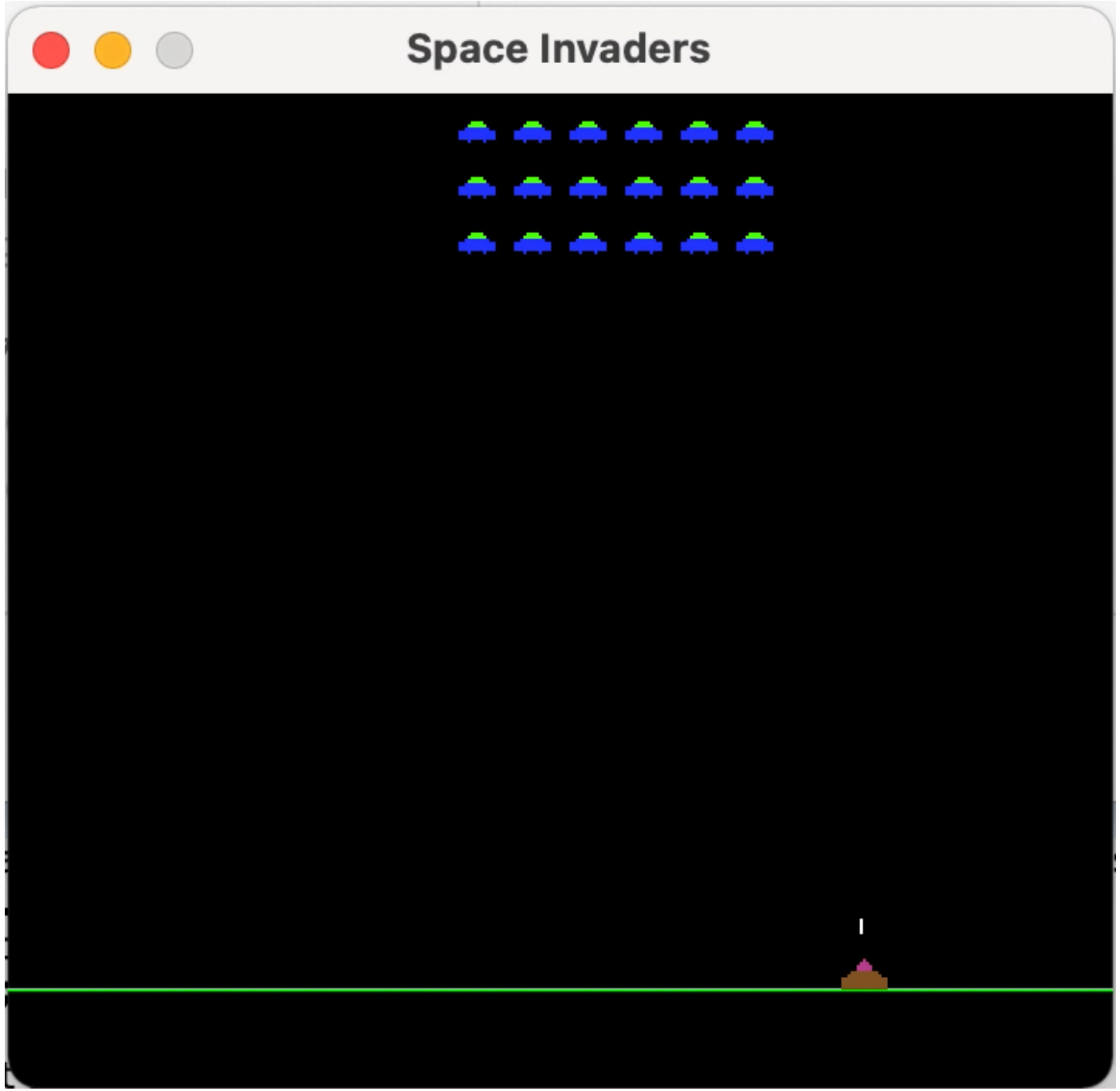
Aula Projeto

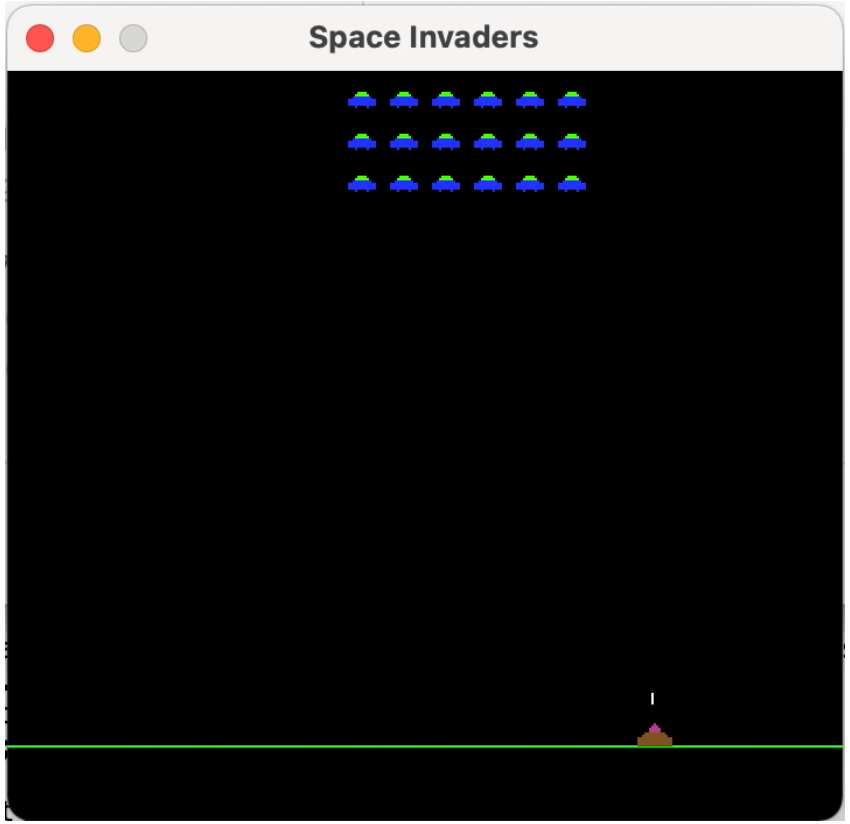
Apresentação do projeto

2022/2023

Evolving a Feed-Forward Neural Network to Control the classic arcade game Space Invaders.



Estado do jogo



Estado do jogo pode ser reprentado por um vector com seguinte informação normalizada:

- Posição dos 18 invasores (x,y)
- Posição das 18 bombas (x,y)
- Posição do jogador (x)
- Posição da bala disparada pelo jogador (x,y, enabled)

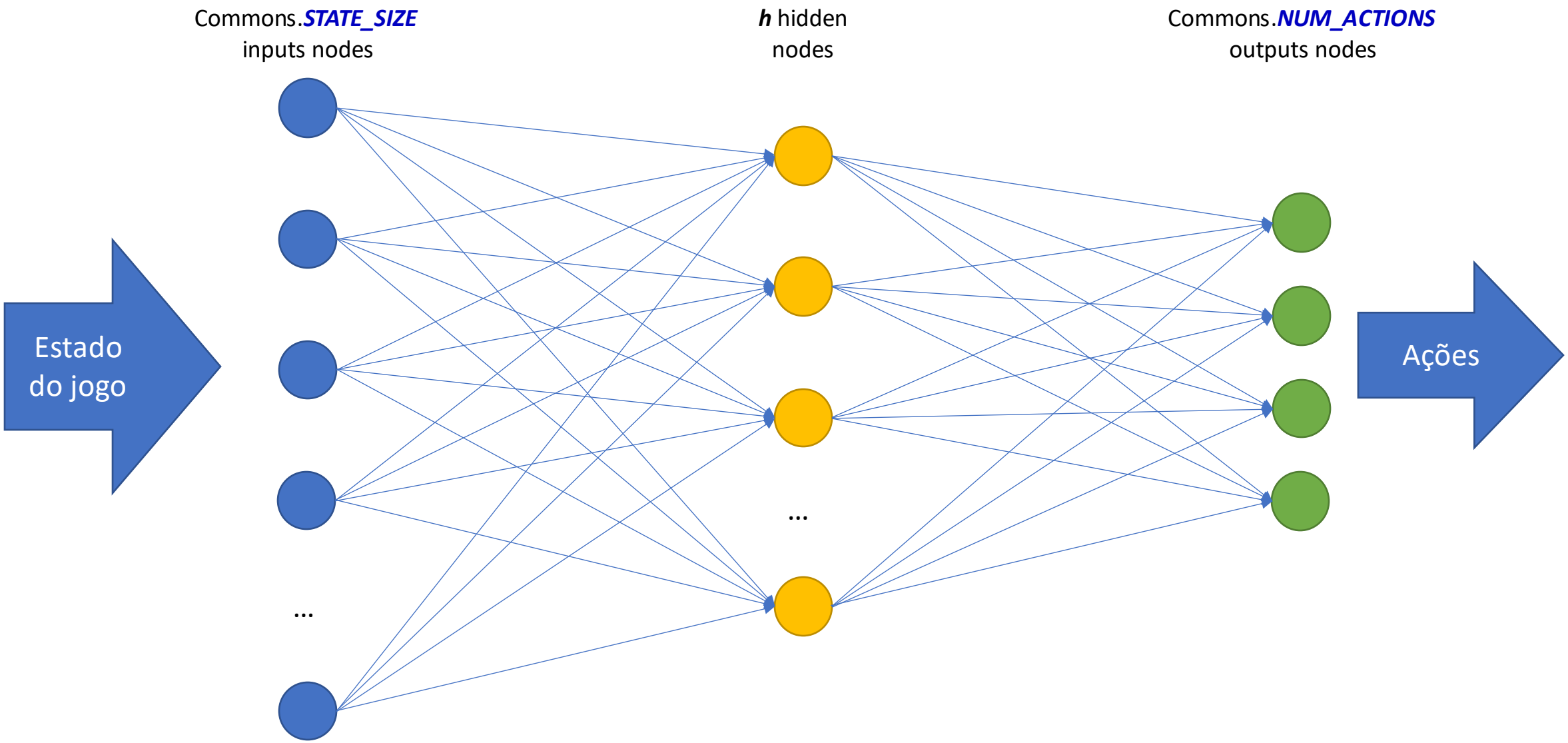
Tamanho do vector:

```
int STATE_SIZE = Commons.NUMBER_OF_ALIENS_TO_DESTROY * 3 * 2 + 1 + 3;

int NUMBER_OF_ALIENS_TO_DESTROY = 18;

//exemplo
Double[] state = {
```

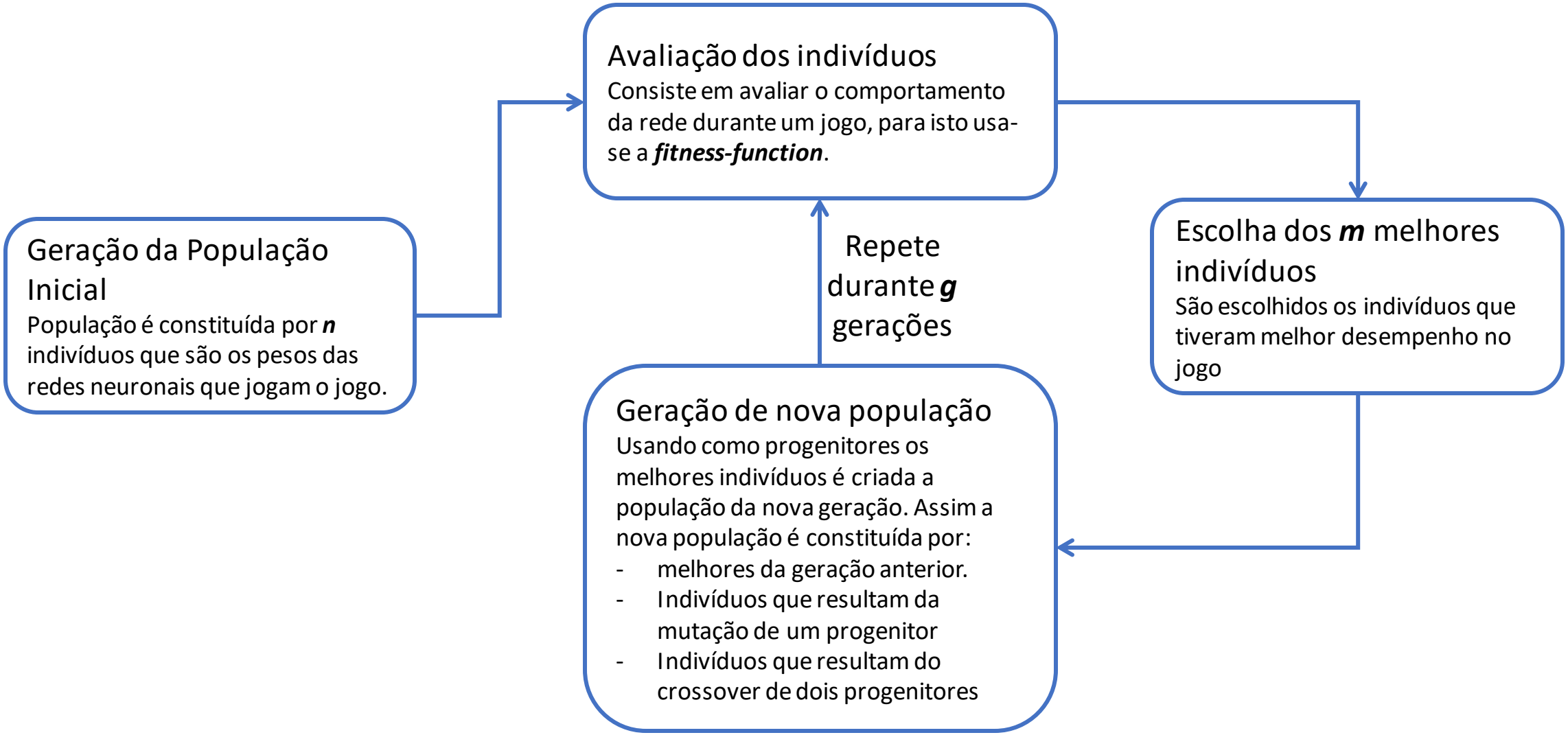
Passos

1. Design and develop a suitable **feed-forward neural network** architecture with an input layer to receive the game state information and an output layer to produce the player's actions.
2. Use a **genetic algorithm to optimize** the network's weights by implementing the evolution process, including selection, mutation, and crossover. The fitness function used will evaluate the network's performance in the game.
3. Generate a population of neural networks and evaluate their performance in the game using the fitness function.
4. Apply the evolutionary process over several generations to improve the network's performance.
5. Summarize the project's results in a **short report** (max three pages), including the design and implementation of the neural network and genetic algorithm, the evolved network's performance, and the network's potential applications for controlling other games and real-world systems.

Feed-Forward Neural Network



Evolutionary algorithm



Space Invaders

To interact with the Space Invaders game, an API is provided that includes the necessary methods.

The Board class constructor creates an instance of the Board class with a game controller.

The **setSeed** method sets the seed value for the random number generator.

The **createState** method creates a normalized state representation of the game board that can be used as input to the controller, such as the neural network.

The **run** method evaluates the fitness of a controller and is the main game loop of the Board class.

The **getFitness** method calculates and returns the fitness score of the game.

Space Invaders

```
public Board(GameController controller) {
        …
}

private double[] createState() {
        double[] state = new double[Commons.STATE_SIZE];
        …
        return state;
}
```

Space Invaders

```
public void run() {
        while (inGame) {
                update();
        }
}

public Double getFitness() {
        double fitness = (double) (getDeaths() * 10000 + getTime());
        return fitness;
}

public void setController(GameController controller) {
        this.controller = controller;
}
```