PONG

Programando o Jogo *Pong* Utilizando Algoritmo Genético

Aluna:
Giordanna De Gregoriis
Professor:
Claudomiro Sales

Agenda

- Introdução;
- Motivação
- Diagramas;
- Genótipo;
- Fitness;
- Crossover;
- Mutação;
- Seleção;
- Testes;
- Resultados;
- Conclusão;
- Referências.

Introdução

- *Pong* simula um tênis de mesa.
- Dois jogadores competem entre si atingindo uma bola com suas respectivas raquetes e mandando-a para o lado oposto.
- Toda vez que rebatida, a bola aumentará de velocidade, até que seja arremessada para fora, assim marcando ponto para o jogador que a arremessou.
- O objetivo do jogo é conseguir a maior pontuação.

Motivação

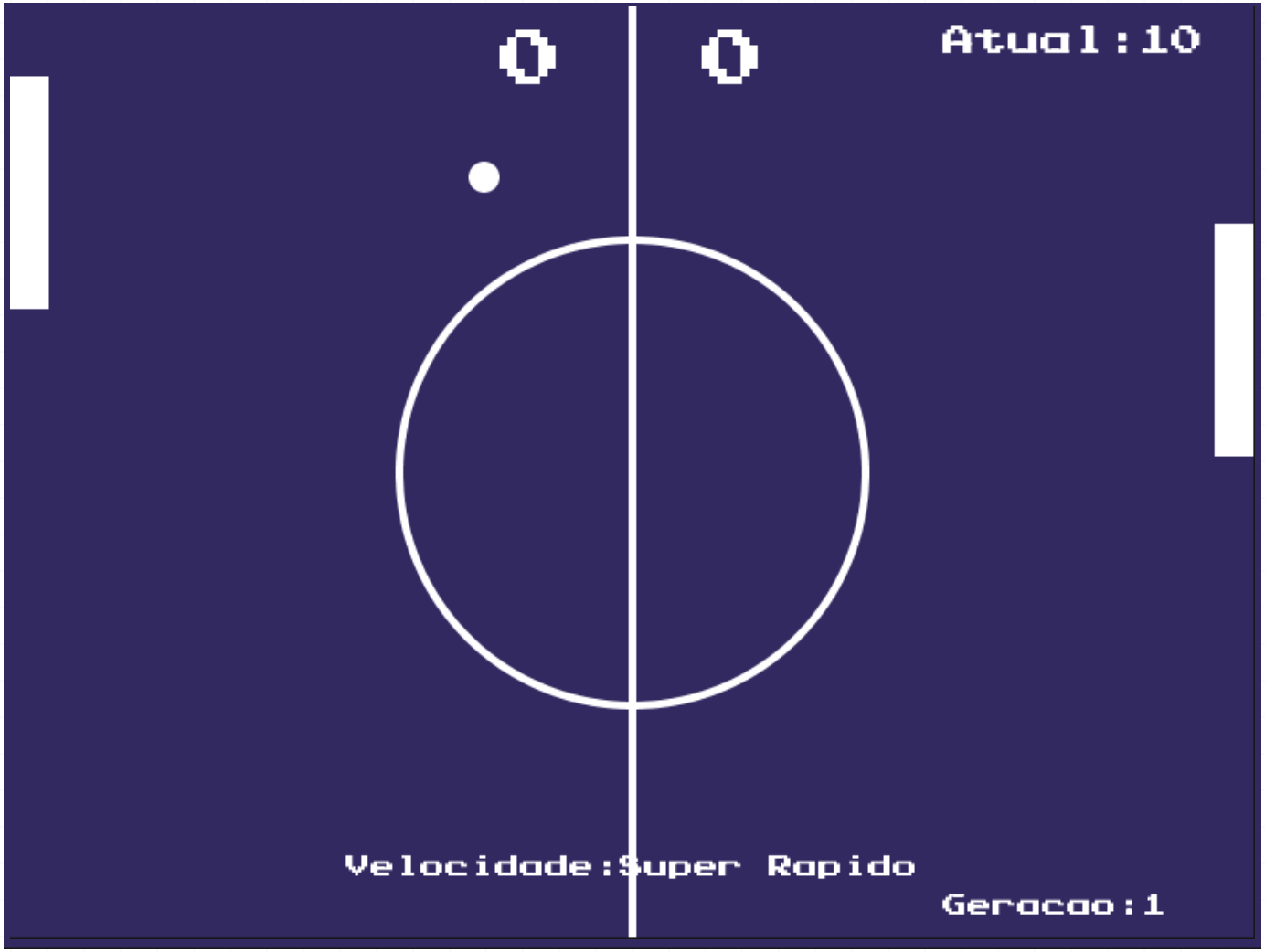
- Conseguir resultados que:

1. Funcionem;
2. Sejam melhores que soluções clássicas de AI para *Pong*.

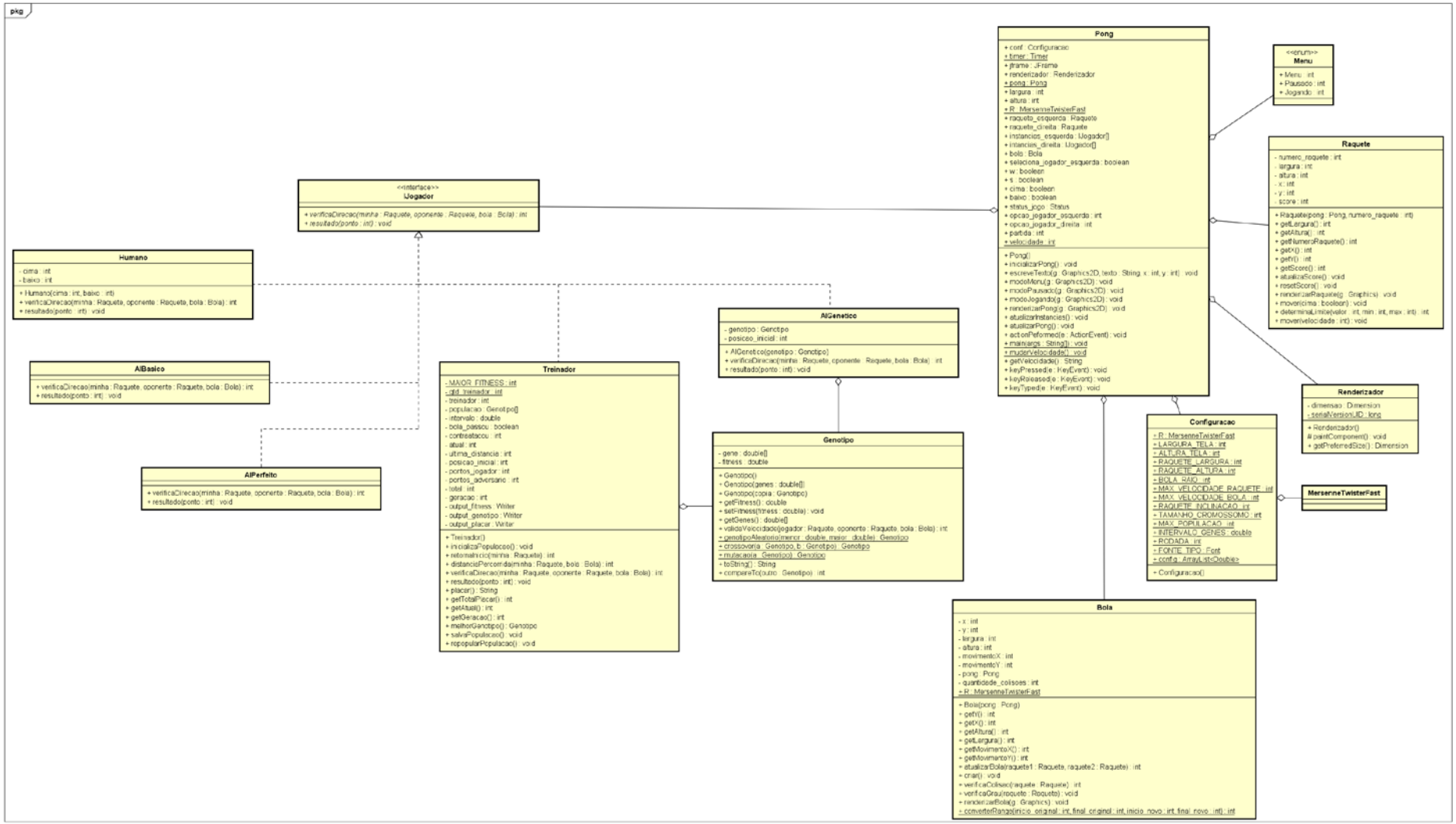
Introdução


PONG GENETICO
<< Jogador 1: AI Basico >>
Jogador 2: AI Treinador
Pressione Space para jogar

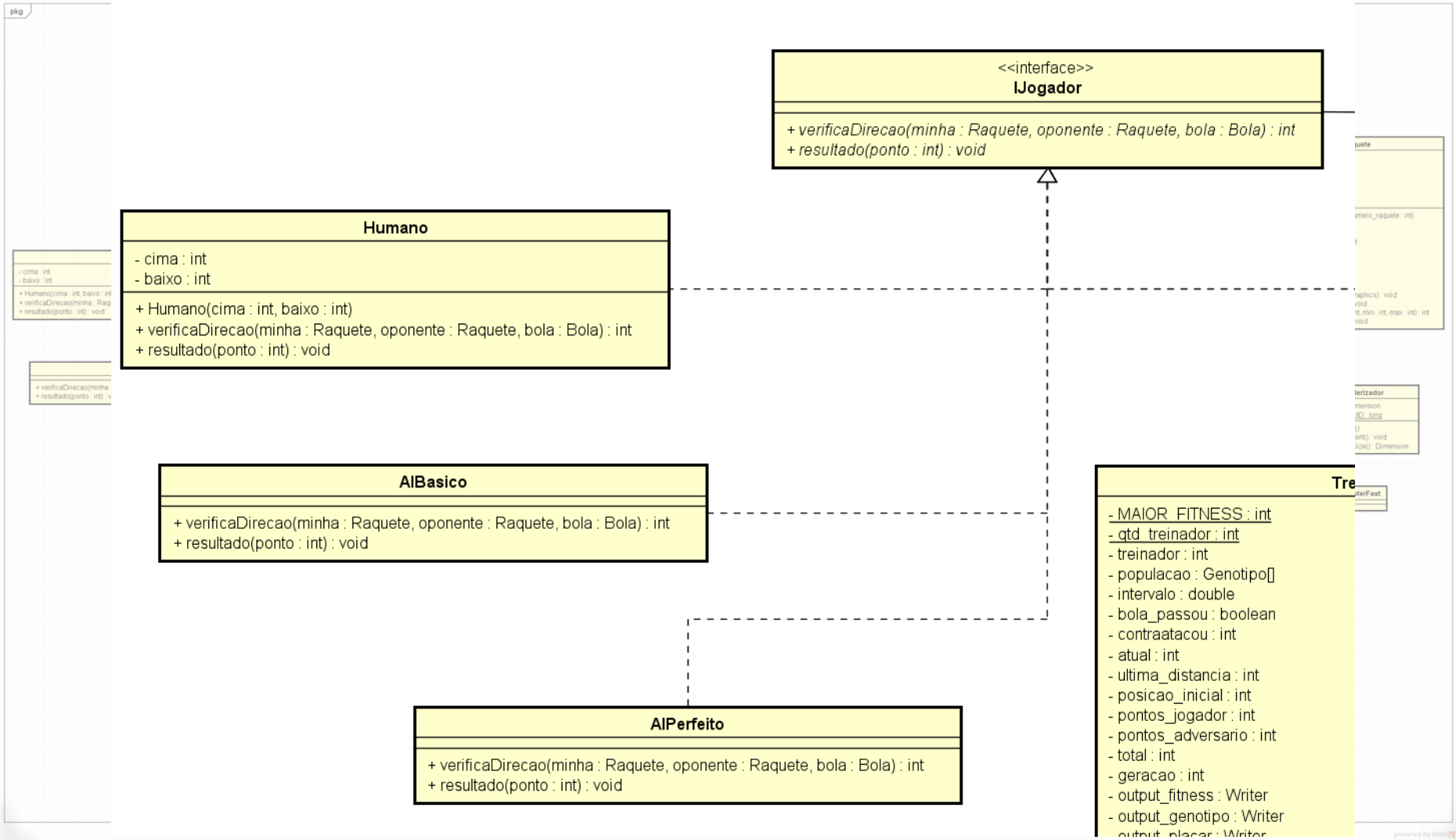
Introdução



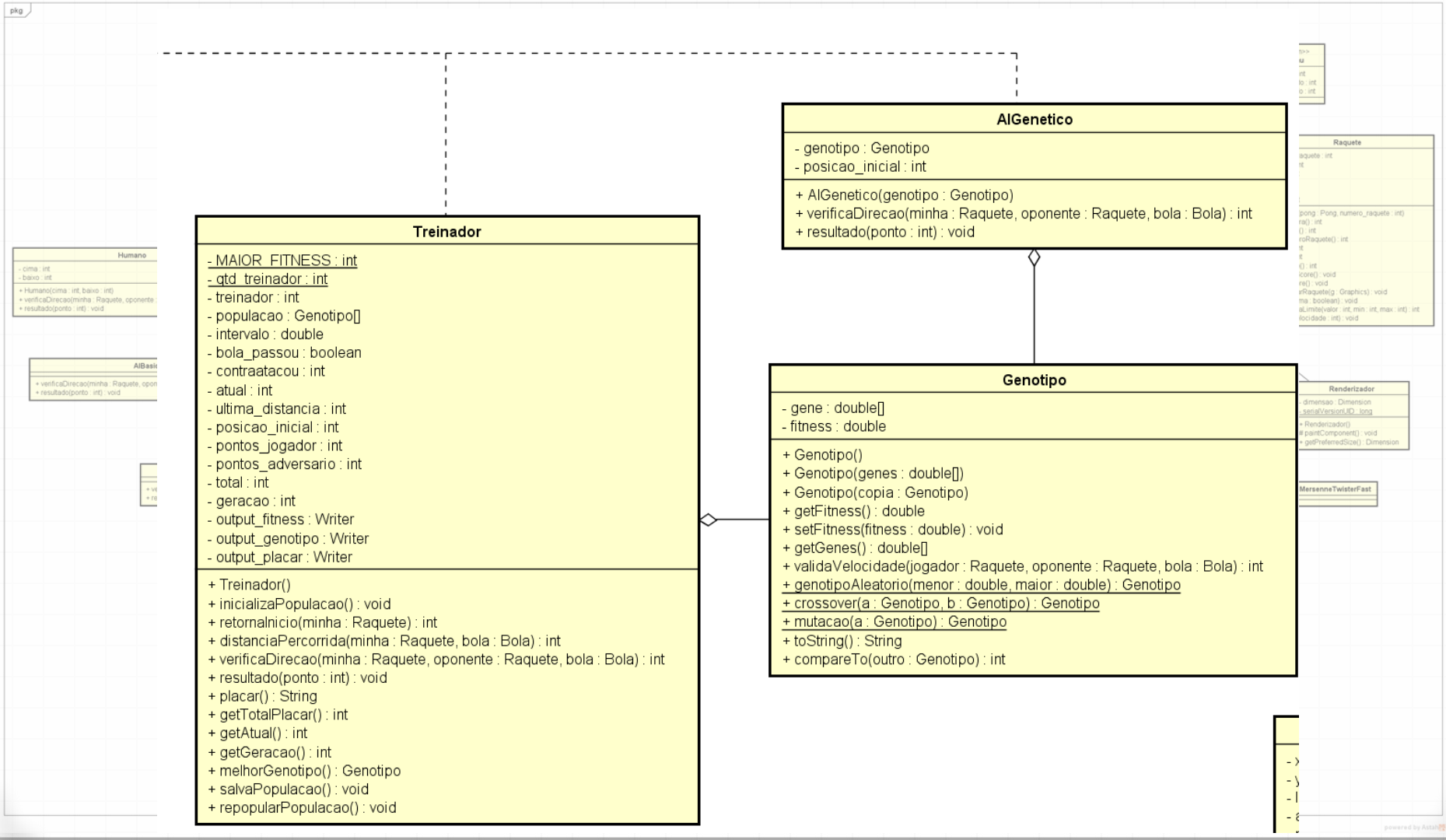
Diagramas - Classes



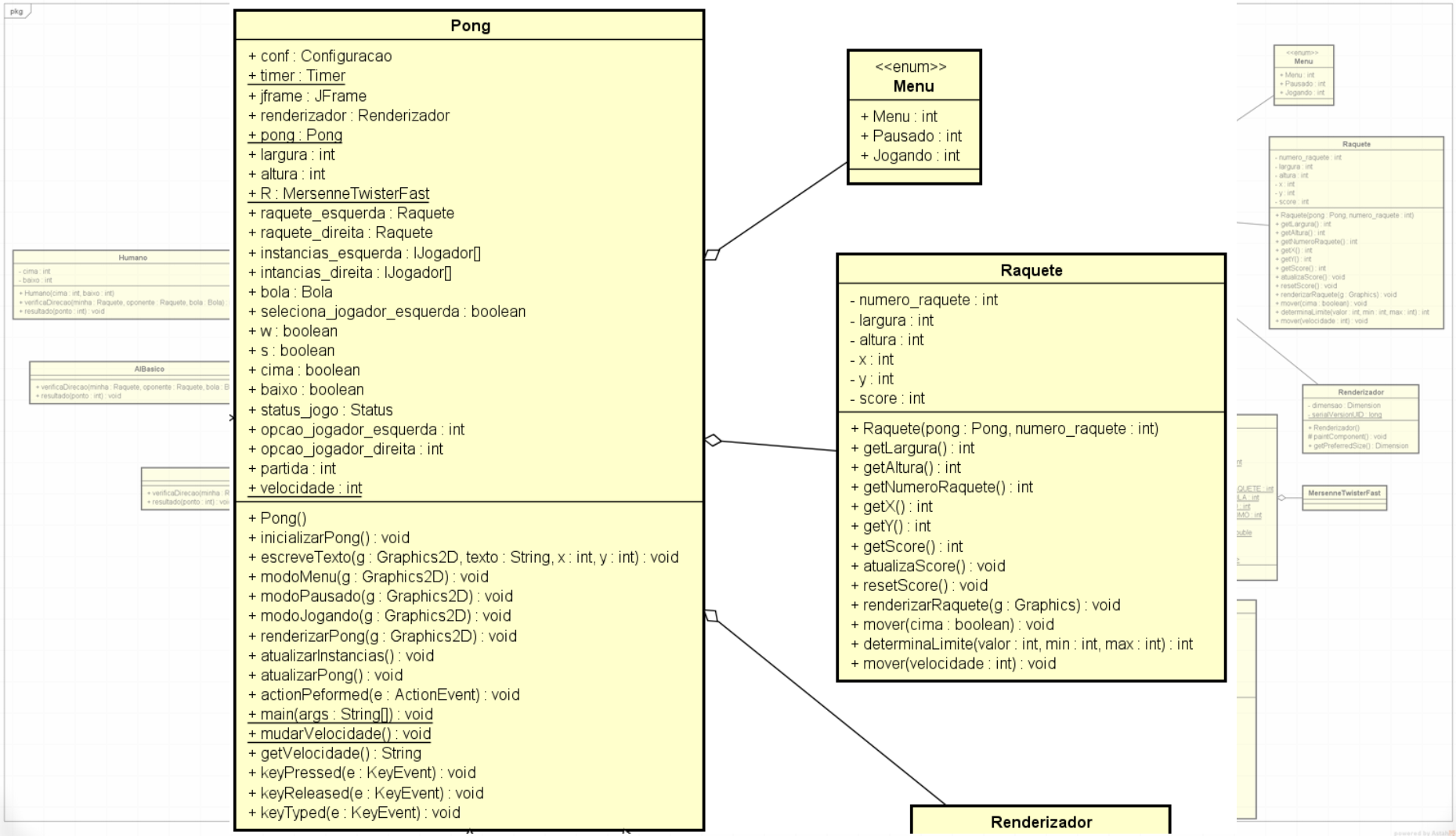
Diagramas - Classes



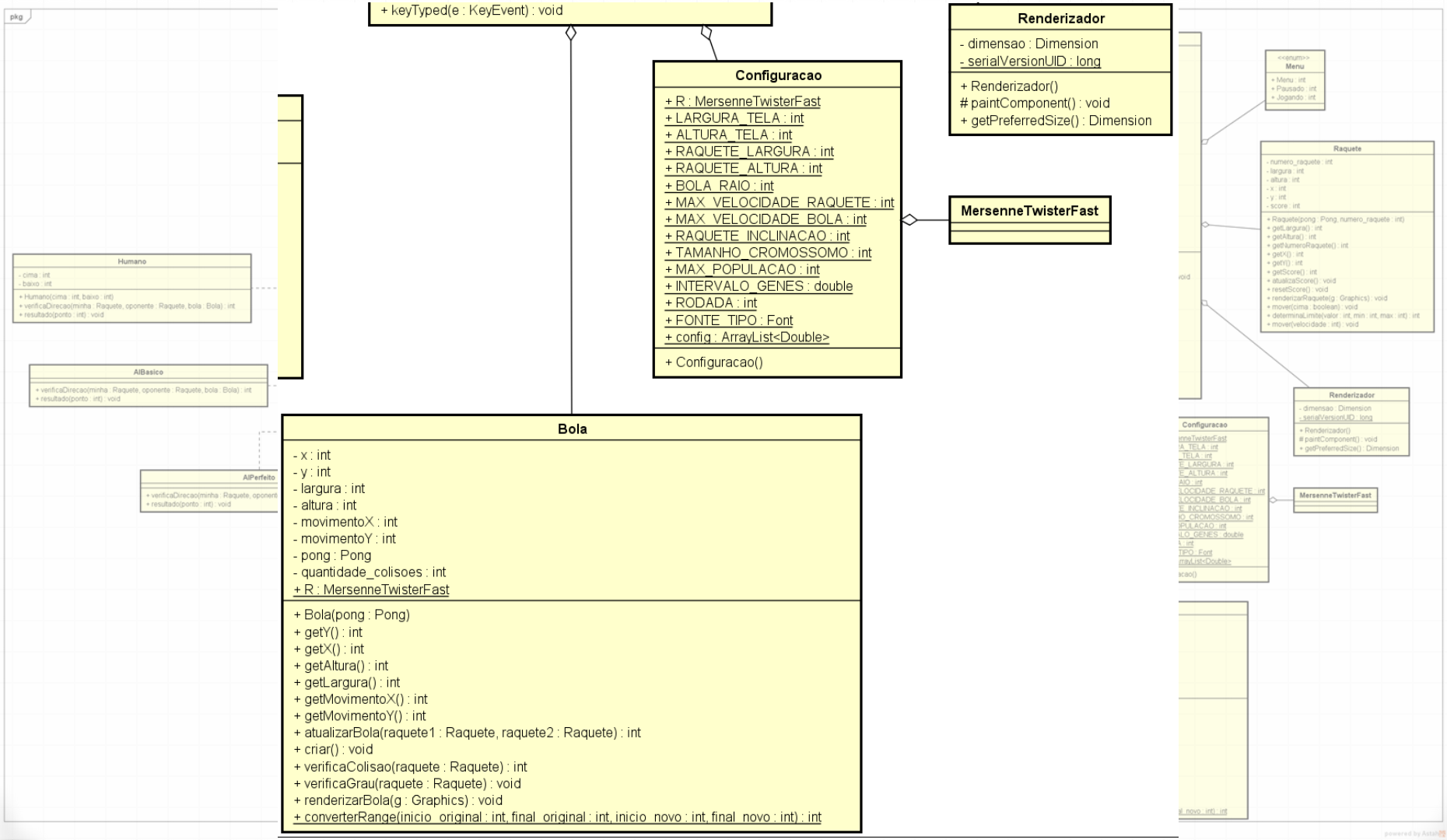
Diagramas - Classes



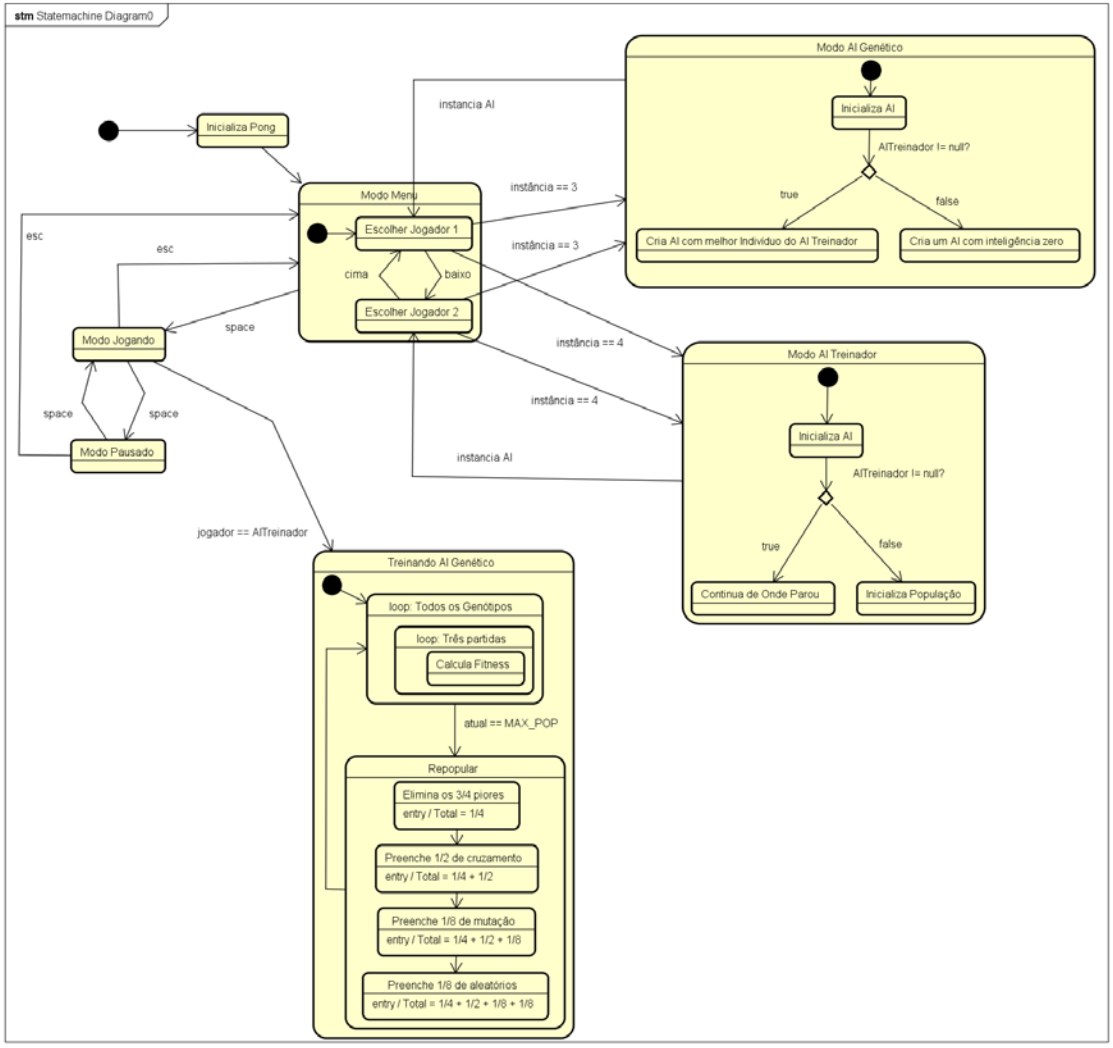
Diagramas - Classes



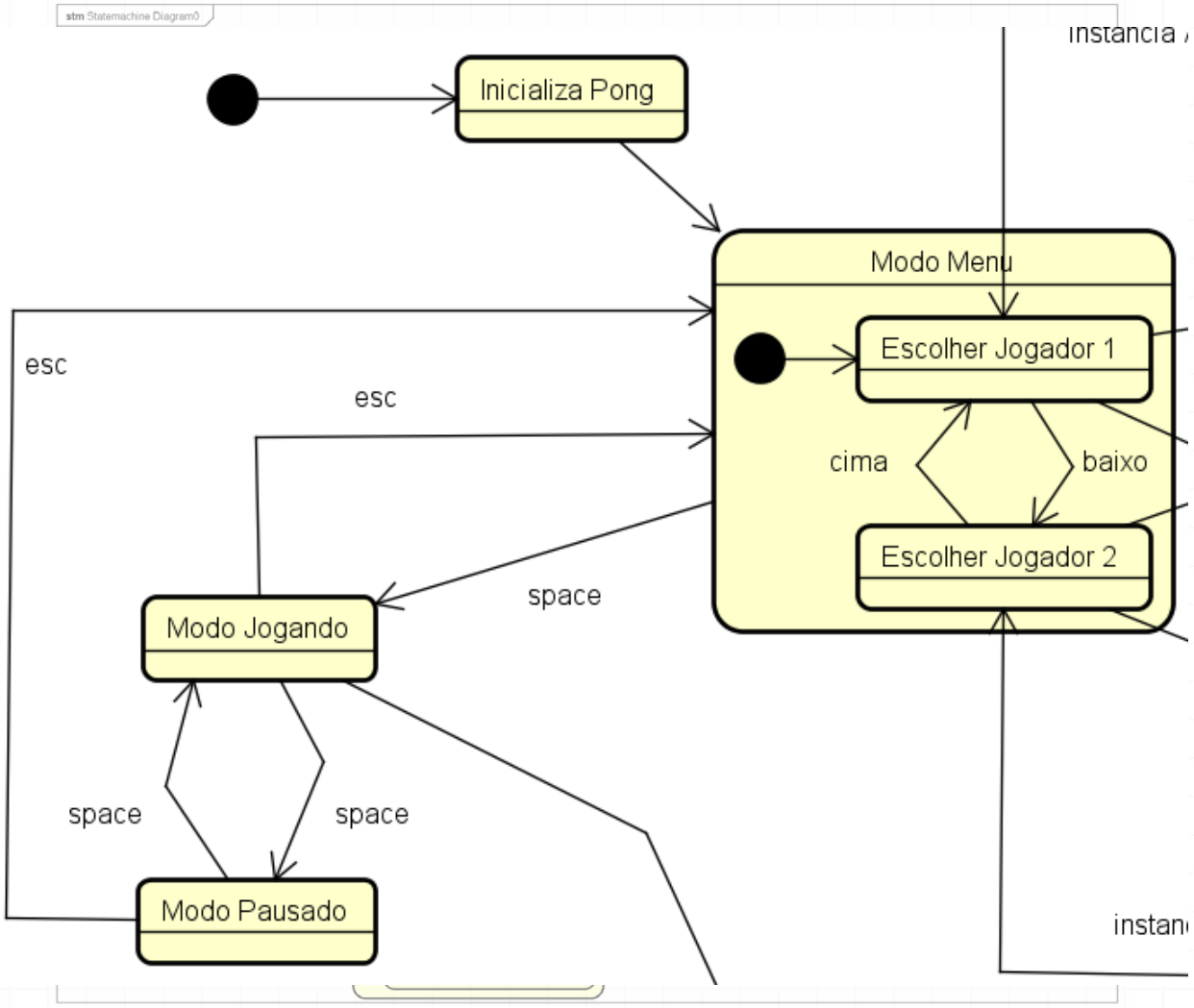
Diagramas - Classes



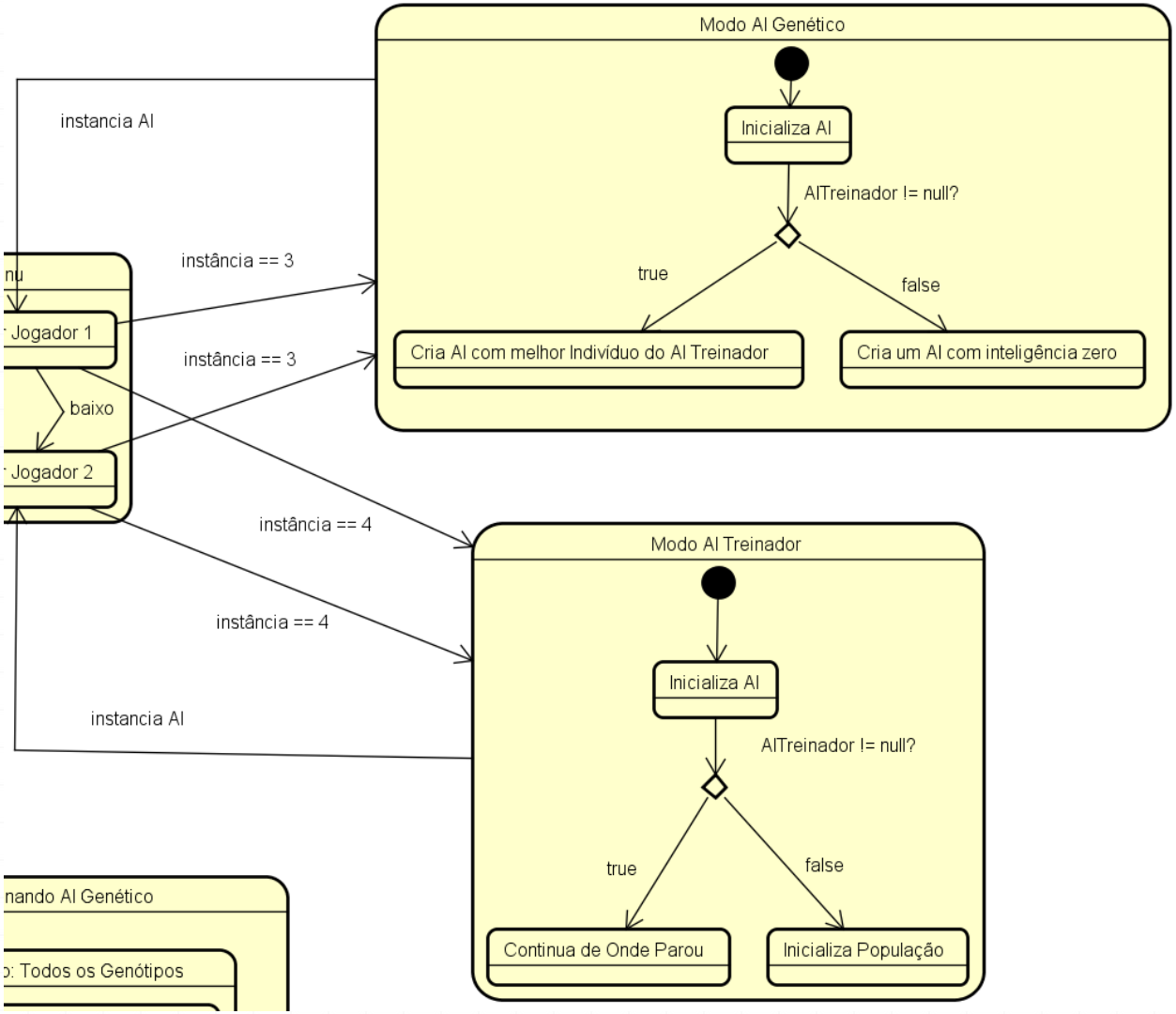
Diagramas - Estados



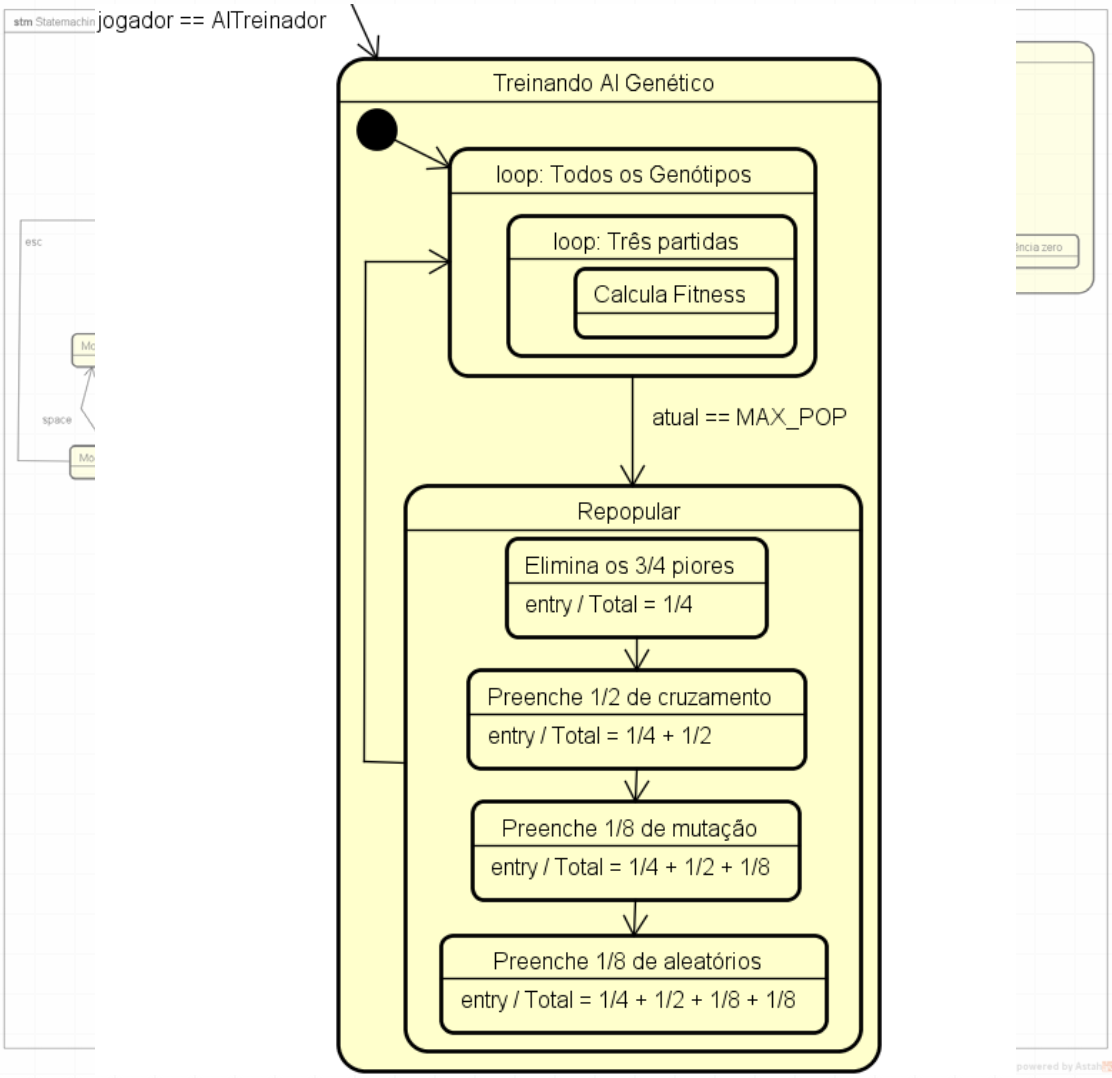
Diagramas - Estados



Diagramas - Estados



Diagramas - Estados



Genótipo

- Constitui-se de uma cadeia de 3 *doubles* que servem como pesos que multiplicam os seguintes valores para calcular a velocidade e direção da raquete:
  - velocidade Y da bola;
  - posição Y da bola;
  - posição Y da raquete do indivíduo.

Genótipo

- Exemplo de um genótipo:

Peso Vel. Y Bola	Peso Pos. Y Bola	Peso Pos. Y Raquete
0,22895	1,063869	-1,12079

Genótipo

- Onde é utilizado os genes:

```
// retorna velocidade da raquete
public int validaVelocidade(Raquete jogador, Raquete oponente, Bola bola){
    return (int) (gene[0] * bola.getMovimentoY() + gene[1] * bola.getY() +
            gene[2] * jogador.getY());
}
```

Fitness

- **Contribui pro Fitness:**
  - Se o jogador marcou algo.
  - Quantas vezes a bola foi rebatida pela raquete.
  - Se o jogador errou, por quanto ele errou.

Fitness

- **Em uma rodada:**
- $Fitness = 9999 \times P + C + (480 - 3 \times D)$
- **Onde:**
  - P = Pode ser 0 ou 1. Se o indivíduo marcou um ponto ou não.
  - C = Quantidade de vezes que conseguiu rebater a bola com a raquete.
  - D = A distância entre a bola e a raquete no momento em que o jogador sofreu um ponto.

Fitness

- Se o indivíduo já possui um fitness "antigo" (da geração anterior ou de rodadas anteriores), tira-se uma média:

- $$Fitness = \frac{9999 \times P + C + (480 - 3 \times D) + \text{Fitness Antigo}}{2}$$

Fitness

```
int fitness = 0;

total++;
if (ponto > 0){
    fitness = MAIOR_FITNESS; // se marcou um ponto então ele é bom mesmo
    pontos_jogador++;
}
else if (ponto < 0){
    pontos_adversario++;
}

fitness += contraatacou;
fitness += 480 - 3 * ultima_distancia;
bola_passou = false;

if (populacao[atual].getFitness() != 0){
    // faz uma média com o fitness antigo
    populacao[atual].setFitness(populacao[atual].getFitness() + fitness)
    populacao[atual].setFitness(populacao[atual].getFitness() / 2);
}
else{
    populacao[atual].setFitness(fitness);
}
```

Crossover

- Uma média aritmética é feita com cada gene dos pais.

```
// retorna filho produzido por dois pais através de crossover por média aritmética
public static Genotipo crossover(Genotipo a, Genotipo b){
    Genotipo novo = new Genotipo();
    for (int i = 0 ; i < Configuracao.TAMANHO_CROMOSSOMO ; i++){
        novo.gene[i] = (a.gene[i] + b.gene[i]) / 2.0;
    }

    return novo;
}
```

Mutação

- É somado um valor aleatório entre -0.1 e 0.1.

```
// mutação do genótipo. para tentar escapar da solução subótima local
public static Genotipo mutacao(Genotipo a){
    Genotipo g = new Genotipo();

    for (int i = 0 ; i < Configuracao.TAMANHO_CROMOSSOMO ; i++){
            g.gene[i] = a.gene[i] + ( -0.1 + 0.2 * Configuracao.R.nextDouble());
    }
    return g;
}
```

Seleção

- Ordena-se a população de acordo com o fitness, do maior para o menor;
- Elimina-se $^{3}/_{4}$ dos piores, deixando os melhores; (25%)
- Preenche-se metade com cruzamento dos melhores genótipos com um aleatório da geração; (50%)
- Adiciona-se $^{1}/_{8}$ de indivíduos aleatórios da população sofrendo mutação; (12,5%)
- Preenche o resto com indivíduos aleatórios. (12,5%)

Seleção

```
// ordena população
Arrays.sort(populacao);

//torna os piores 3/4 nulos
for (int i = populacao.length/4 ; i < populacao.length ; i++){
    populacao[i] = null;
}

int j = 0;
int outro;
// preenche metade com os melhores + um genótipo aleatório da mesma geração
for (int i = populacao.length/4 ; i < 3*populacao.length/4 ; i++){
    while (true){
        outro = Configuracao.R.nextInt(populacao.length/4);
        if (j != outro) break;
    }
    populacao[i] = new Genotipo(Genotipo.crossover(populacao[j], populacao[outro]));
    j++;
}
```

Seleção

```
// adiciona alguns poucos genótipos com mutação
for (int i = 3*populacao.length/4 ; i < 7*populacao.length/8 ; i++){
    outro = Configuracao.R.nextInt(populacao.length/2-1);
    populacao[i] = new Genotipo(Genotipo.mutacao(populacao[outro]));
}

// preenche o resto com novos genótipos aleatórios
for (int i = 7*populacao.length/8 ; i < populacao.length ; i++){
    populacao[i] = Genotipo.genotipoAleatorio(-intervalo, intervalo);
}
```

Testes

- Foram realizados dois testes:
  - AI Treinador *versus* AI Básico;
  - AI Treinador *versus* AI “Perfeito”.

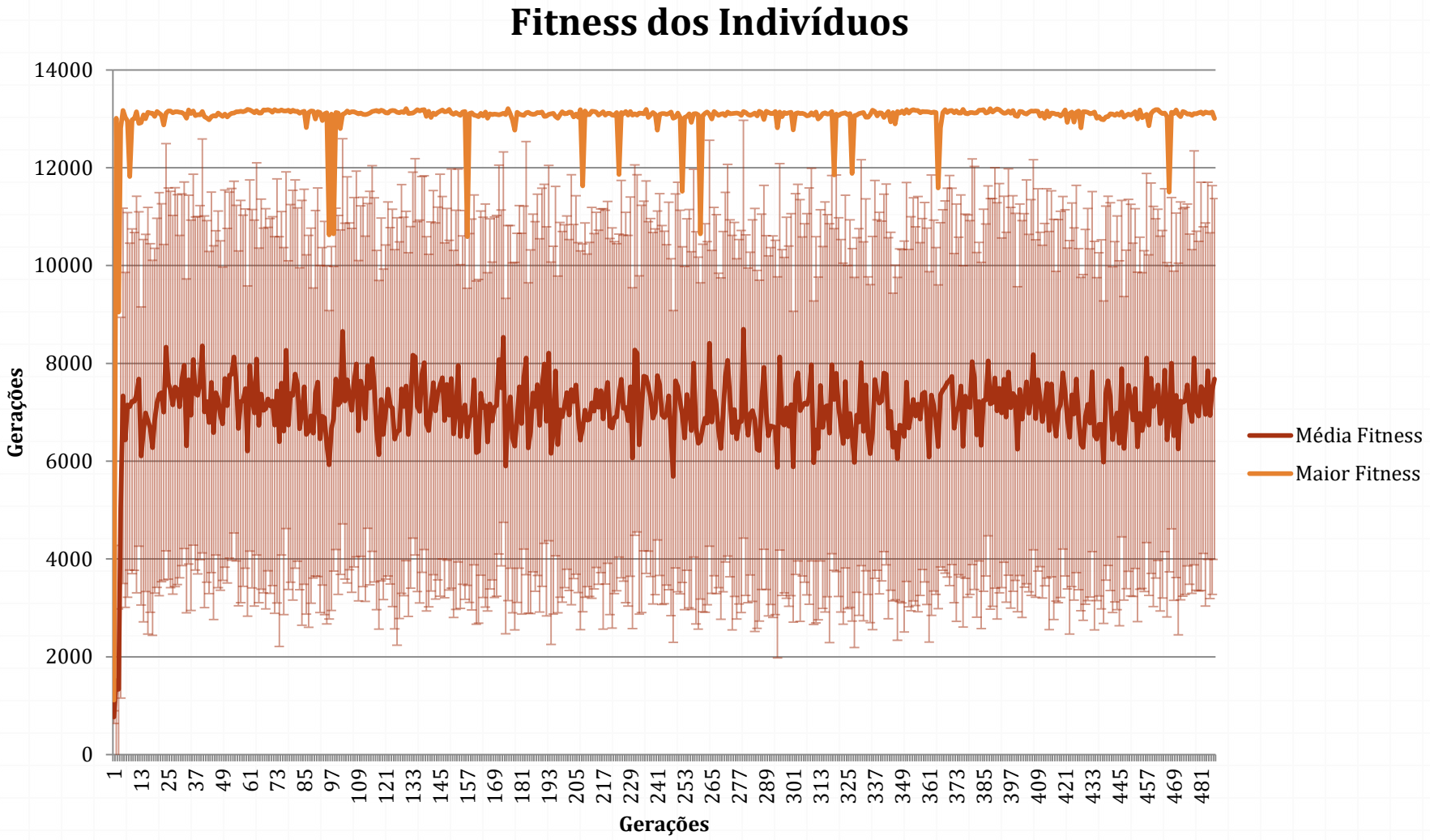
- População: 30 indivíduos;

Resultados

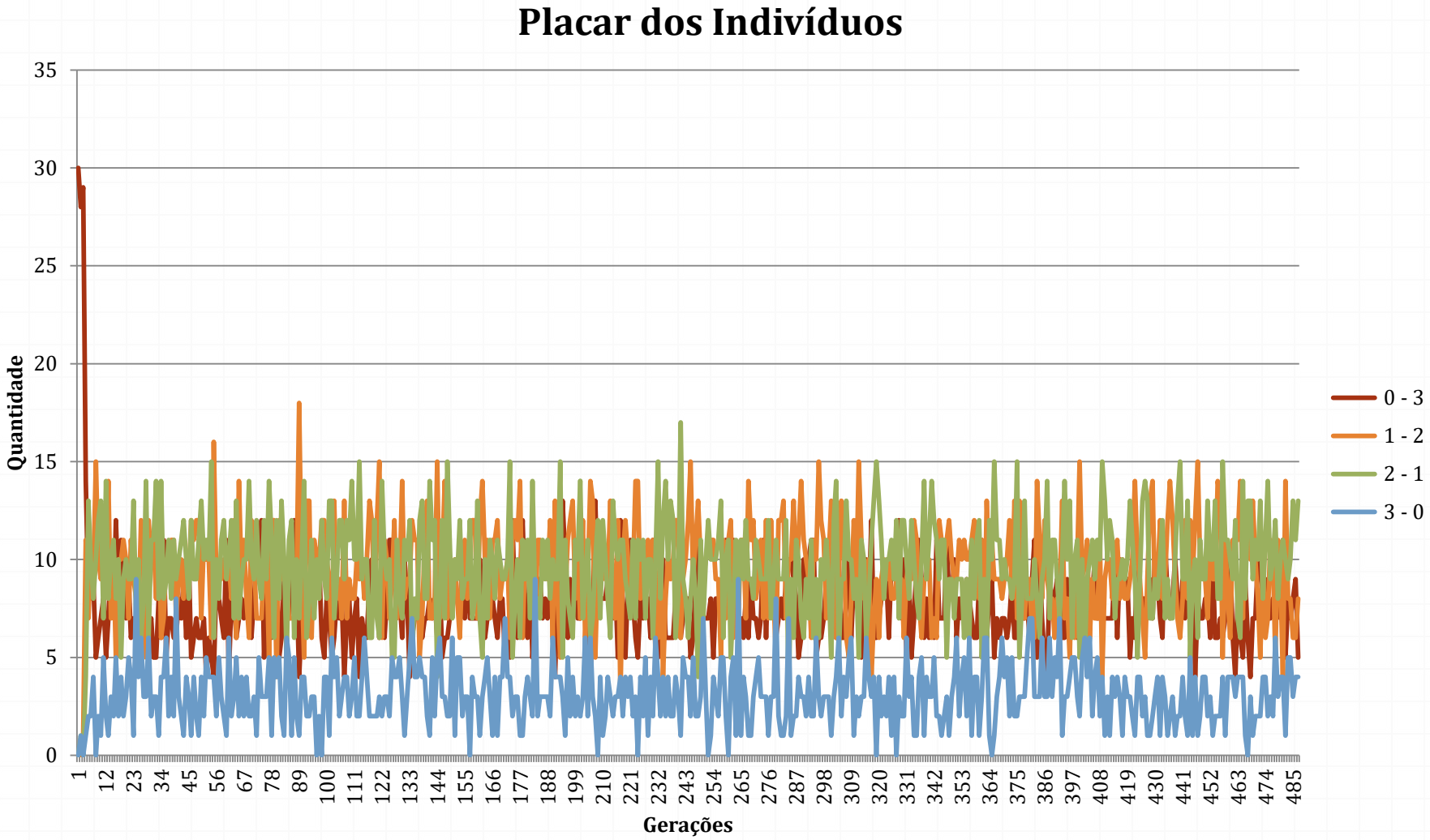
AI Treinador *versus* AI Básico

Gerações: 487

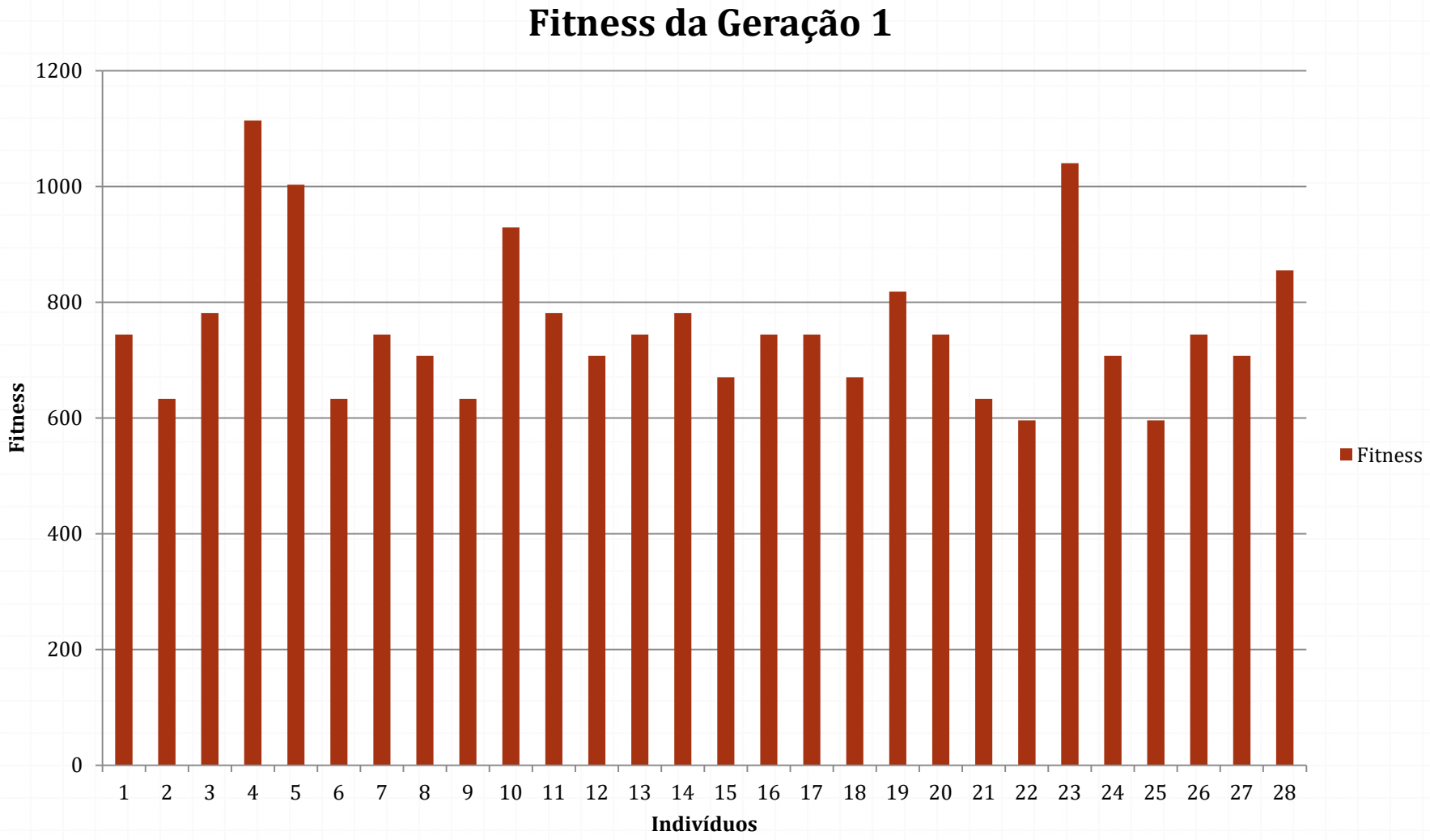
Resultados


Fitness dos Indivíduos
14000
12000
10000
8000
6000
4000
2000
0
Gerações
1
13
25
37
49
61
73
85
97
109
121
133
145
157
169
181
193
205
217
229
241
253
265
277
289
301
313
325
337
349
361
373
385
397
409
421
433
445
457
469
481
Gerações
Média Fitness
Maior Fitness

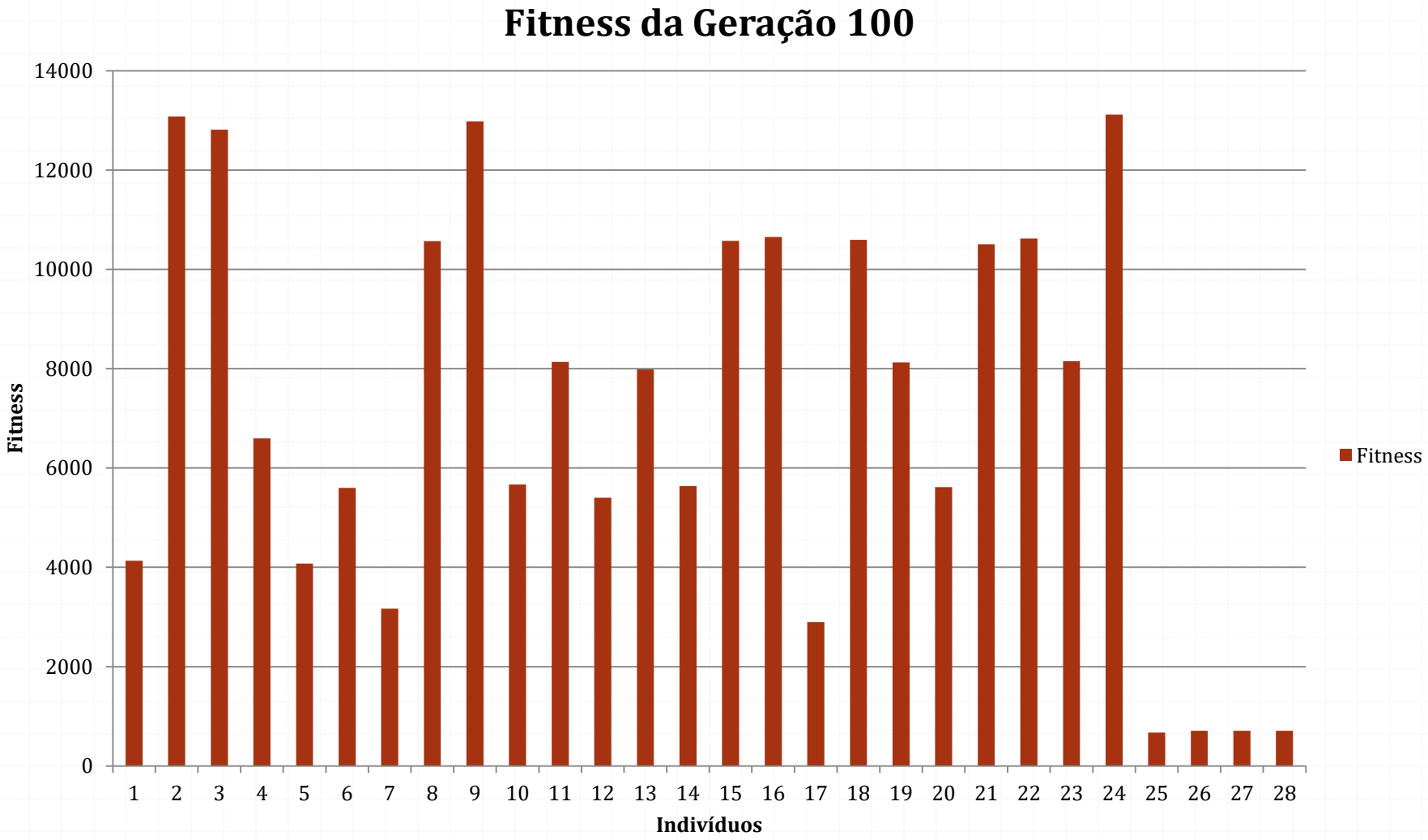
Resultados


Placar dos Indivíduos
Quantidade
35
30
25
20
15
10
5
0
1
12
23
34
45
56
67
78
89
100
111
122
133
144
155
166
177
188
199
210
221
232
243
254
265
276
287
298
309
320
331
342
353
364
375
386
397
408
419
430
441
452
463
474
485
Gerações
0 - 3
1 - 2
2 - 1
3 - 0

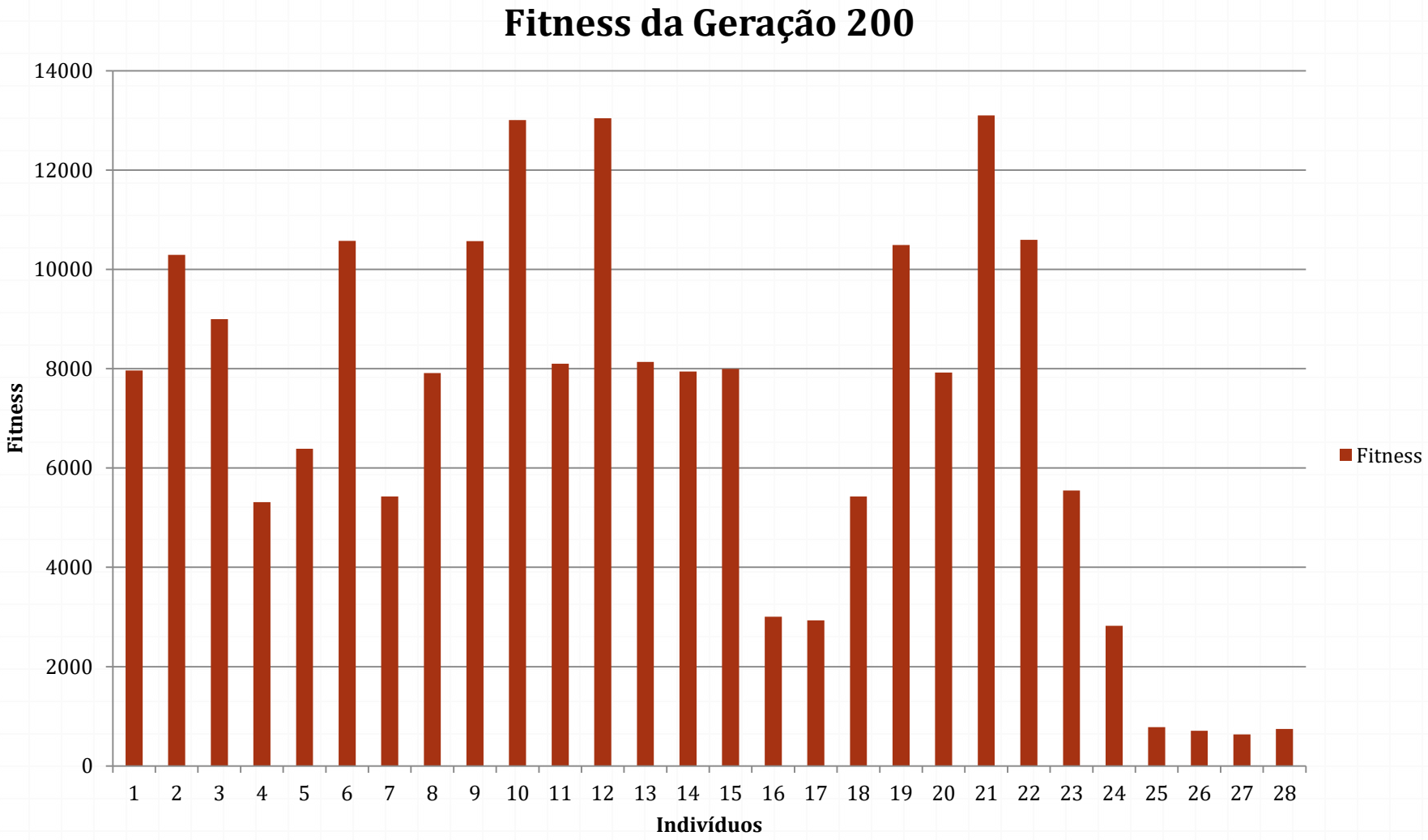
Resultados


Fitness da Geração 1
1200
1000
800
600
400
200
0
Fitness
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
Indivíduos
Fitness

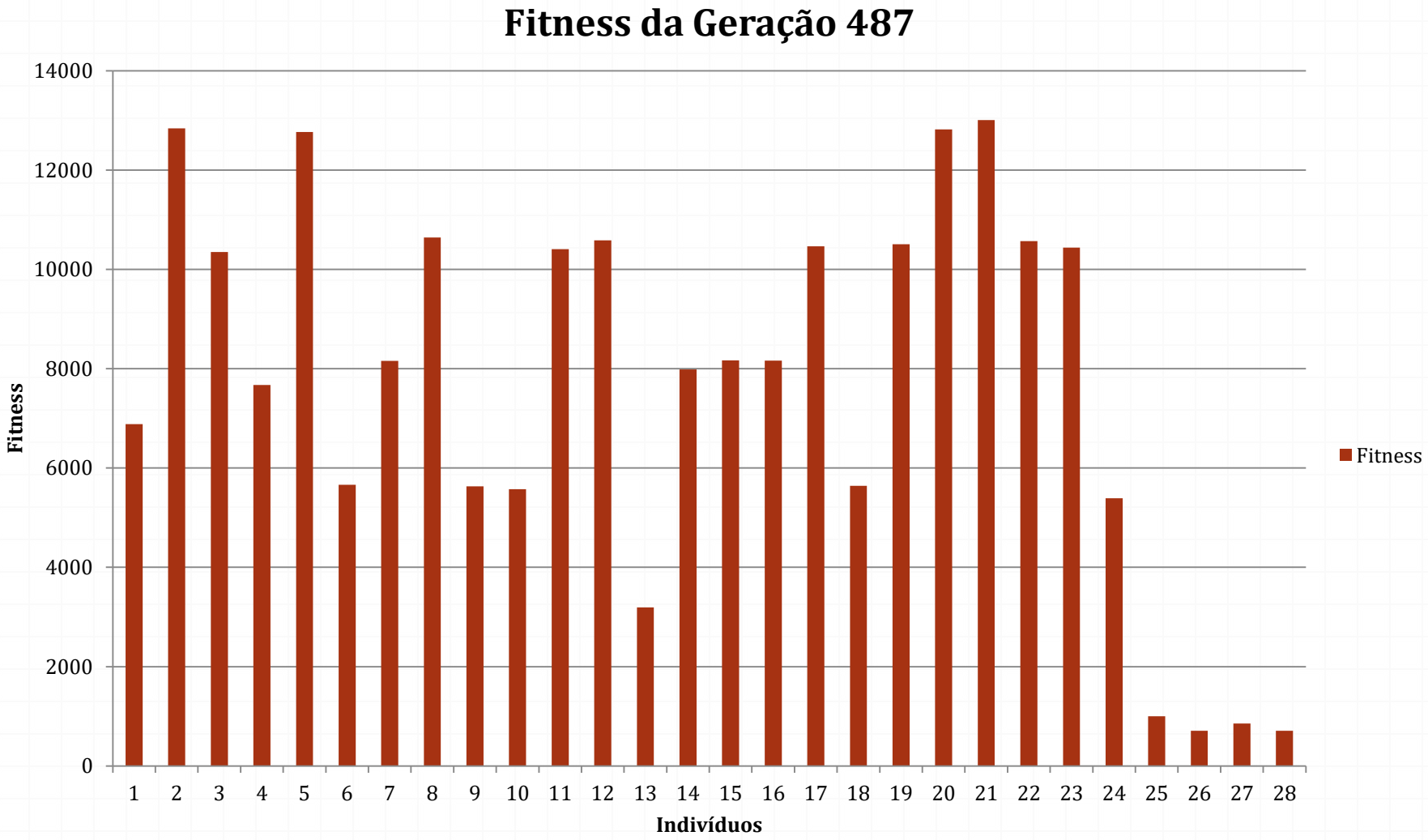
Resultados


Fitness da Geração 100
14000
12000
10000
8000
6000
4000
2000
0
Fitness
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
Indivíduos
Fitness

Resultados


Fitness da Geração 200
14000
12000
10000
8000
6000
4000
2000
0
Fitness
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
Indivíduos
Fitness

Resultados


Fitness da Geração 487
14000
12000
10000
8000
6000
4000
2000
0
Fitness
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
Indivíduos
Fitness

Resultados

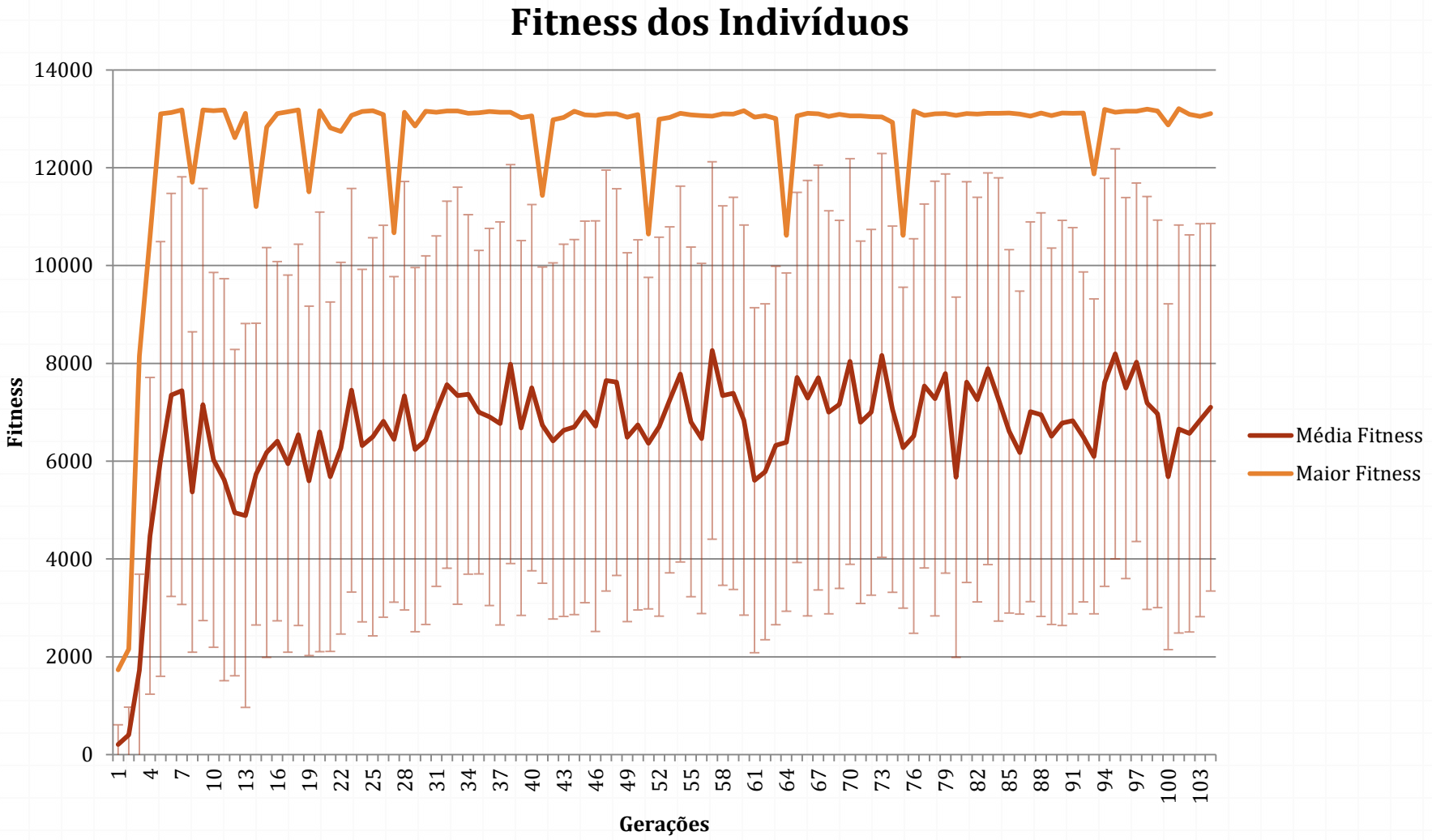
Melhor Genótipo	Peso Vel. Y Bola	Peso Pos. Y Bola	Peso Pos. Y Raquete
Geração 1	-0,20934	-0,90338	-0,33093
Geração 100	0,523905	0,862016	-0,92503
Geração 200	0,43204	0,643573	-0,73053
Geração 487	0,194921	0,973477	-1,12359

Resultados

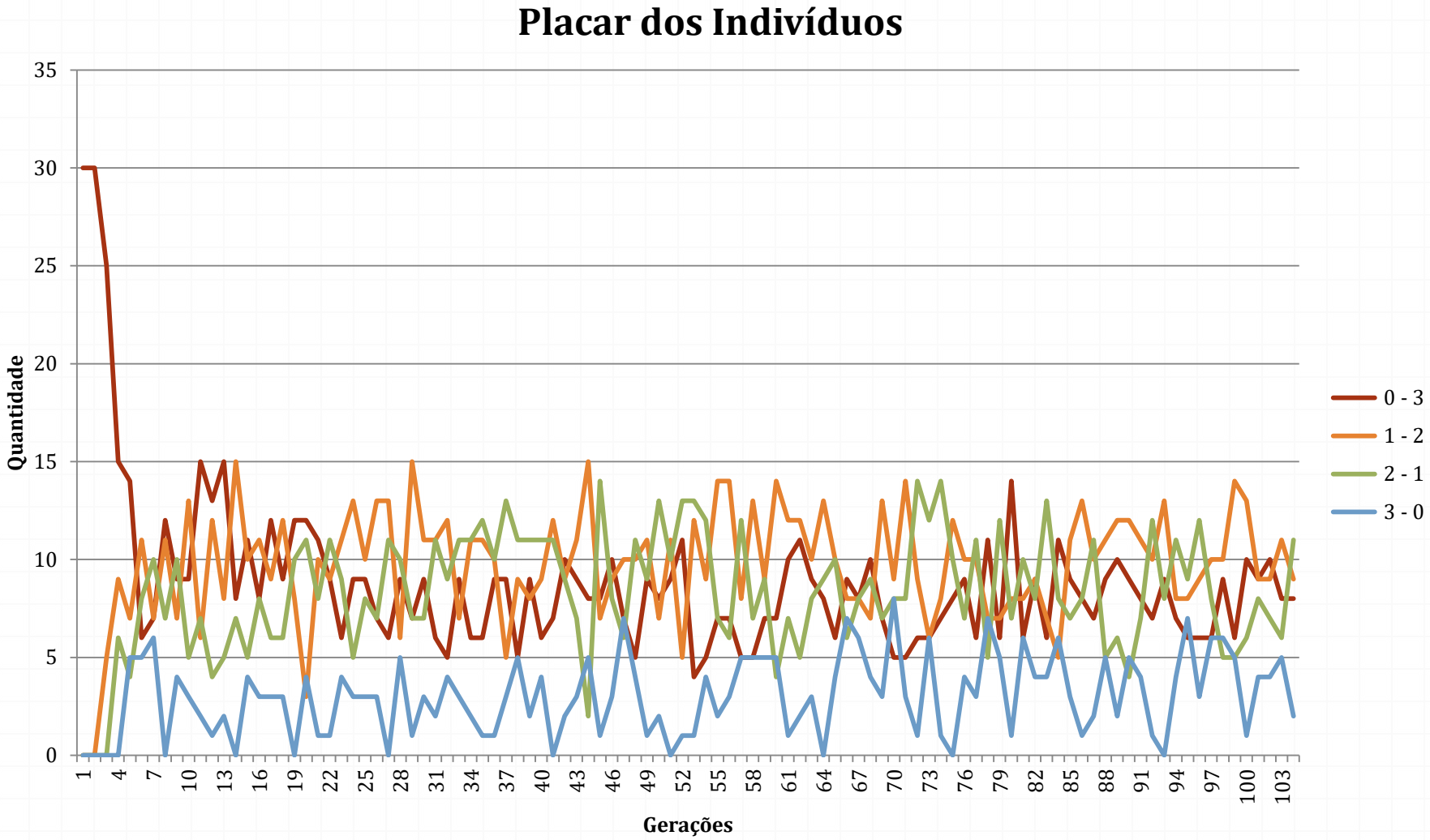
AI Treinador *versus* AI “Perfeito”

Gerações: 104

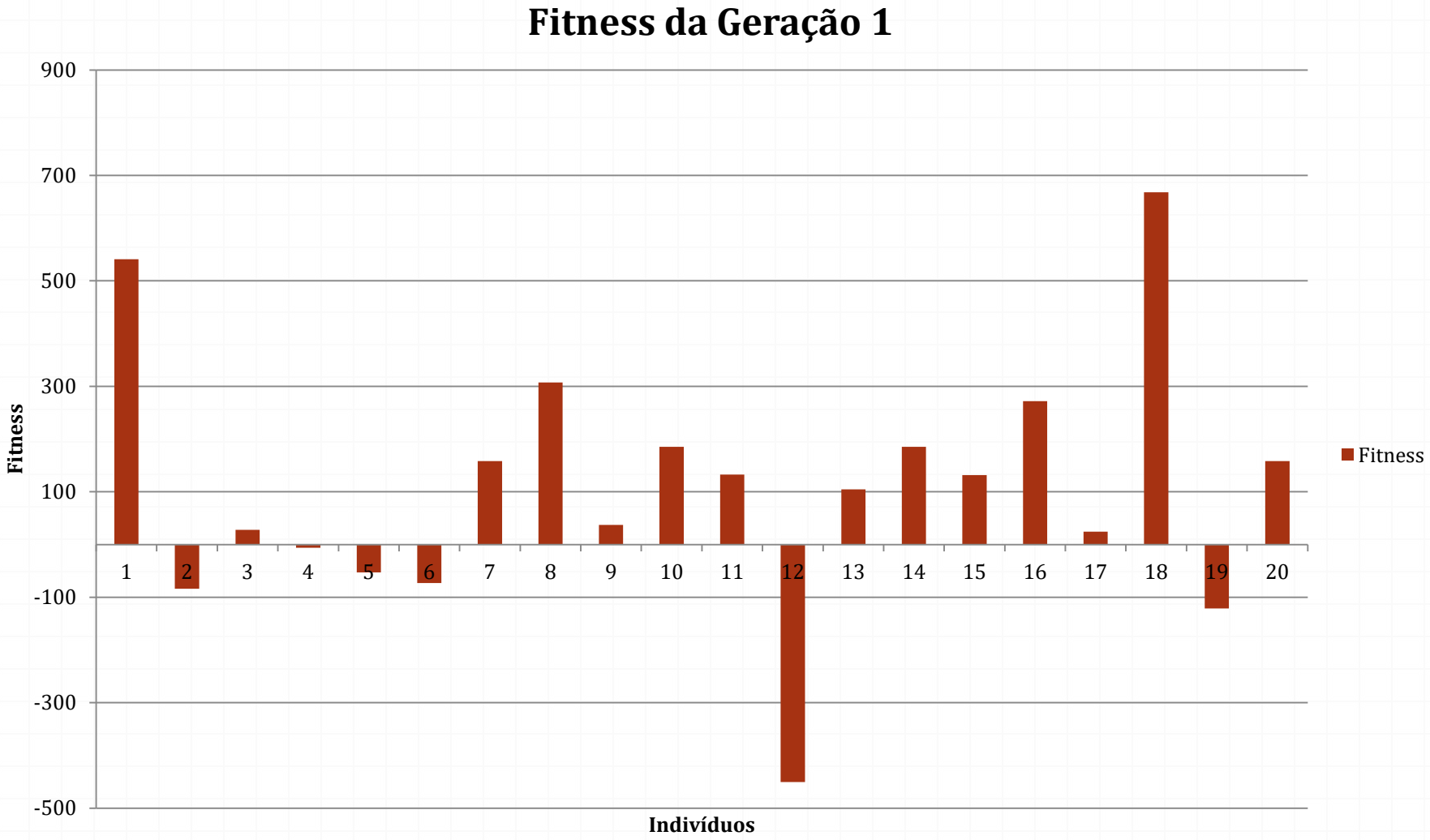
Resultados


Fitness dos Indivíduos
Fitness
14000
12000
10000
8000
6000
4000
2000
0
1
4
7
10
13
16
19
22
25
28
31
34
37
40
43
46
49
52
55
58
61
64
67
70
73
76
79
82
85
88
91
94
97
100
103
Gerações
Média Fitness
Maior Fitness

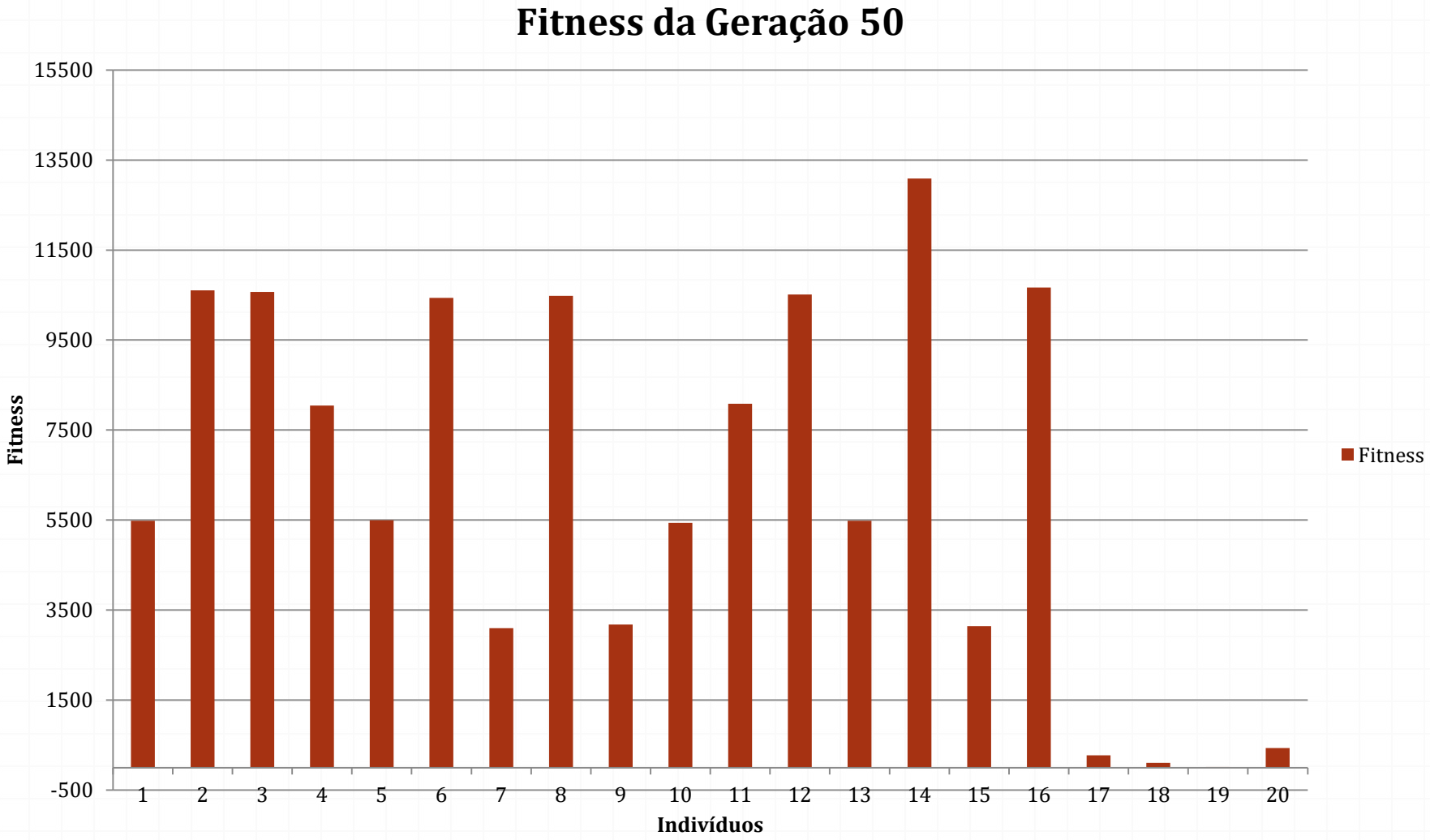
Resultados


Placar dos Indivíduos
Quantidade
35
30
25
20
15
10
5
0
1
4
7
10
13
16
19
22
25
28
31
34
37
40
43
46
49
52
55
58
61
64
67
70
73
76
79
82
85
88
91
94
97
100
103
Gerações
0 - 3
1 - 2
2 - 1
3 - 0

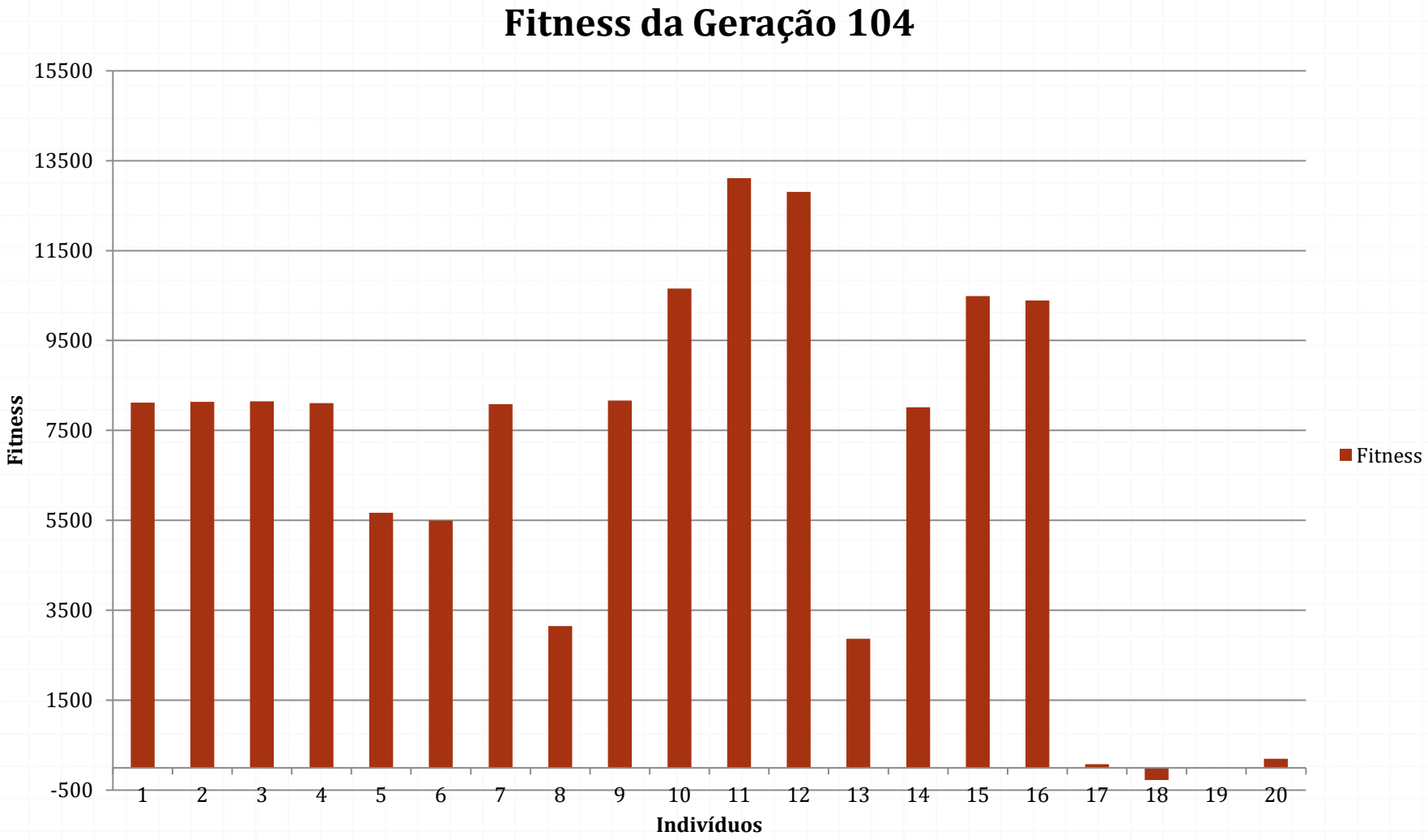
Resultados


Fitness da Geração 1
900
700
500
300
100
-100
-300
-500
Fitness
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Indivíduos
Fitness

Resultados


Fitness da Geração 50
15500
13500
11500
9500
7500
5500
3500
1500
-500
Fitness
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Indivíduos
Fitness

Resultados


Fitness da Geração 104
15500
13500
11500
9500
7500
5500
3500
1500
-500
Fitness
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Indivíduos
Fitness

Resultados

Melhor Genótipo	Peso Vel. Y Bola	Peso Pos. Y Bola	Peso Pos. Y Raquete
Geração 1	-0,54384	0,789705	-0,75384
Geração 50	-0,40925	0,739908	-0,88266
Geração 104	-0,27574	0,844741	-0,90563

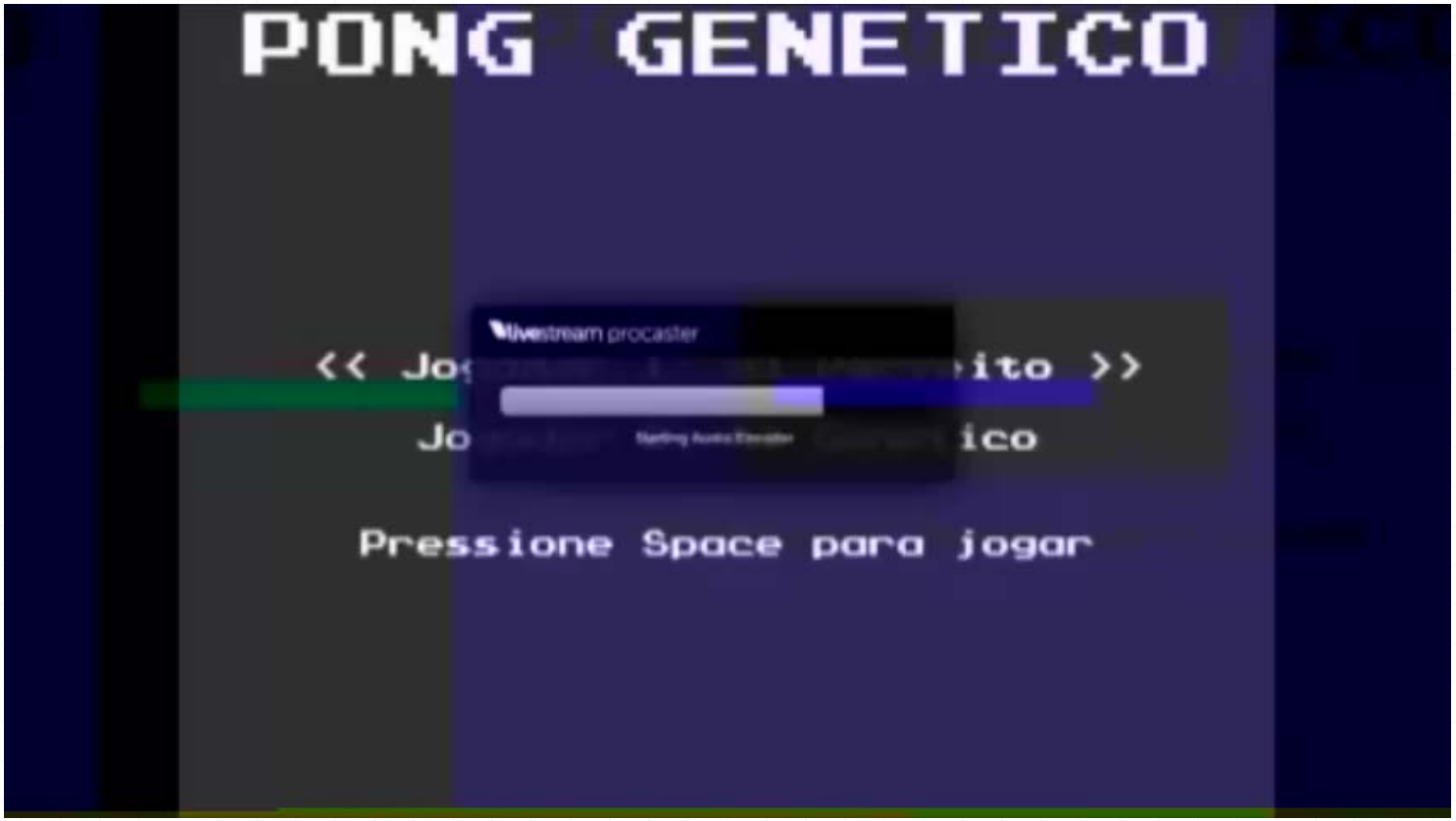
Resultados

- Comparação:

Melhor Genótipo	Peso Vel. Y Bola	Peso Pos. Y Bola	Peso Pos. Y Raquete
AI Perfeito	-0,27574	0,844741	-0,90563
AI Básico	0,194921	0,973477	-1,12359

Resultados

Resultados



Conclusões

- O algoritmo converge extremamente rápido, em torno de 10 gerações necessárias para se obter um bom resultado;
- Ao ser treinado com o AI que faz previsões da posição da bola, o melhor genótipo tende a ficar com o peso em relação a posição Y da bola negativo;
- Já ao ser treinado com o AI que apenas verifica a posição Y da bola, este mesmo peso tende a ser positivo;
- O que foi treinado com o AI Perfeito tende a ganhar mais do que foi treinado com o AI Básico.

Conclusões

- O AI Genético que foi treinado com o AI Básico tende a ganhar mais vezes do AI Básico, pois o algoritmo de calculo da velocidade do AI Genético é uma versão melhorada do AI Básico;
- Já o AI Genético treinado com AI Perfeito tende a perder mais do AI Perfeito, porém consegue vencê-lo.

Referências

- BUSTARD, J. (2014) Programming Pong in Java! (Full Tutorial). Disponível em: <https://youtu.be/1wD2CdFlDaE>.

- FOSTER, T. (2015) Genetic Pong – The Public Var. Disponível em: <http://publicvar.wikidot.com/post:genetic-pong>.

OBRIGADA!