Sistemas de inteligencia artificial
TP2: Algoritmos Genéticos

Integrantes:
- ➔ Lucas Catolino
- ➔ Matias Ricarte

Problema

Mediciones en un reactivo, donde los valores de cada entrada ξ_k , devuelve el valor ζ_k , k = 1, 2, 3.

Se debe hallar X tal que minimice el error:

$$E(W, w, w_0) = \sum_{\mu=1}^{3} (\zeta^\mu - F(W, w, \xi^\mu))^2$$

$$\xi^1 = \begin{pmatrix} 4{,}4793 \\ -4{,}0765 \\ -4{,}0765 \end{pmatrix} \qquad \zeta^1 = 0$$

$$\xi^2 = \begin{pmatrix} -4{,}1793 \\ -4{,}9218 \\ 1{,}7664 \end{pmatrix} \qquad \zeta^2 = 1$$

$$\xi^3 = \begin{pmatrix} -3{,}9429 \\ -0{,}7689 \\ 4{,}8830 \end{pmatrix} \qquad \zeta^3 = 1$$

Donde: $F(W, w, w_0, \xi) = g(\sum_{i=1}^{2} W_j g(\sum_{k=1}^{3} w_{jk}\xi_k - w_{j0}) - W_0) \qquad \xi \in \Re^3$ y $g(x) = \frac{e^x}{1+e^x}$

$$X = (W_0, W_1, W_2, w_{11}, w_{12}, w_{13}, w_{21}, w_{22}, w_{23}, , w_{01}, w_{02})$$

Dificultades encontradas

- Entendimiento del problema
- Errores menores en el enunciado

Implementación y consideraciones

Implementación y consideraciones

Definimos nuestro individuo como un vector de 11 números reales a partir del cual se pueden calcular los valores de la función F y el valor del error E

La población X comienza con valores aleatorios entre -1 y 1

```java
public class Individuo implements Comparable<Individuo> {
    /*
     ** W = (W0, W1, W2) [1x3]
     ** w ==> S [2x3]
     ** w0 ==> R [1x2]
     ** X = (W, S, R) = (W0, W1, W2, w11, w12, w13, w21, w22, w23, w01, w02) [1x11]
     */
    private static final int SIZE = 11;
    private double[] X;
    private double fitness;
    private double[] F3 = new double[3];
```

```java
@Override
public int compareTo(Individuo other) {
    return Double.compare(this.fitness, other.getFitness());
}
```

En el programa utilizamos como función de fitness a -1•E(X) para simplificar comparaciones

Implementación y consideraciones

Hicimos uso del Factory pattern para la implementación de los diferentes métodos de selección y cruzamiento

```
public interface Selector {
    List<Individuo> selectFrom(List<Individuo> inputPopulation,
    int generationCount);
}
```

```
CutOff cutOff = new CutOff(this.params);
Selector selector = new SelectorFactory().getSelector(this.params);
Cross crosser = new CrossFactory().getCross(this.params);
```

```
switch (params.getCrossType()) {
    case SIMPLE:
        return new SimpleCross(params.getI
        ;
    case MULTIPLE:
        return new MultipleCross(params.get
        , params.getKCross());
    case UNIFORM:
        return new UniformCross(params.get
```

Implementación y consideraciones

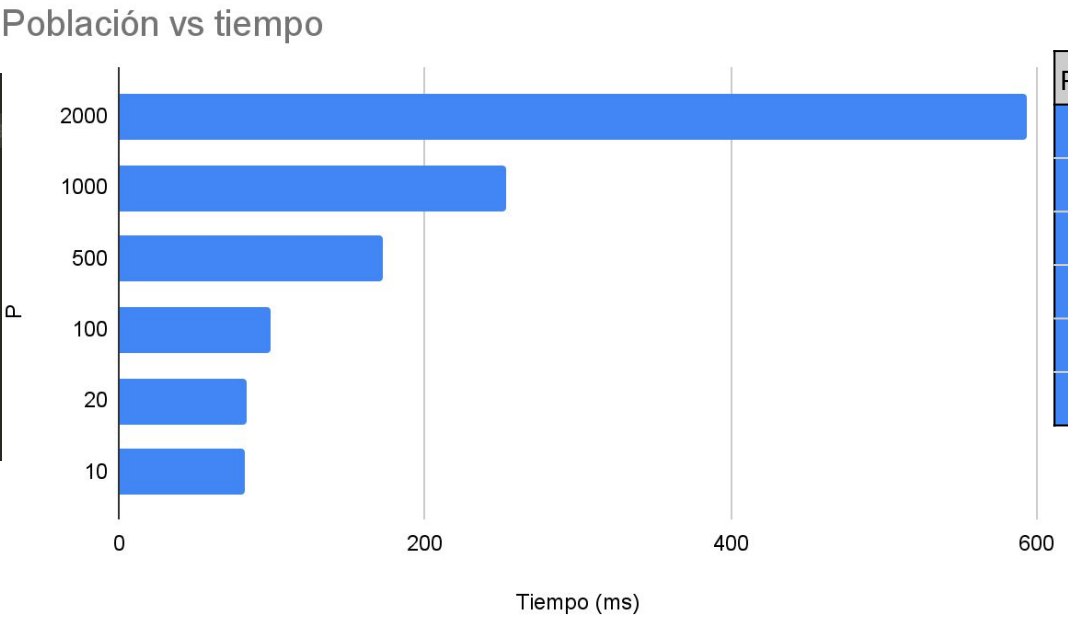
Dado que los valores de los elementos de X son números reales sin (a primera vista) un rango que los delimite, usamos criterios de parada.

- Cantidad máxima de generaciones: 500
- Condición de mínimo fitness (solución aceptable)
- Repetición de estructura X entre una generación y otra
- Poca variación de fitness entre una generación y otra

Consideraciones

Elección del P

```
population: 10
selector.type: stochastic
mutation.deviation: 0.1
mutation.probability: 0.2
cross.type: simple
cutoff.maxgen: 500
cutoff.minacceptable: 0.000000001
cutoff.maxrepstruct: 10
cutoff.maxrepcontent: 5
dump: true
```

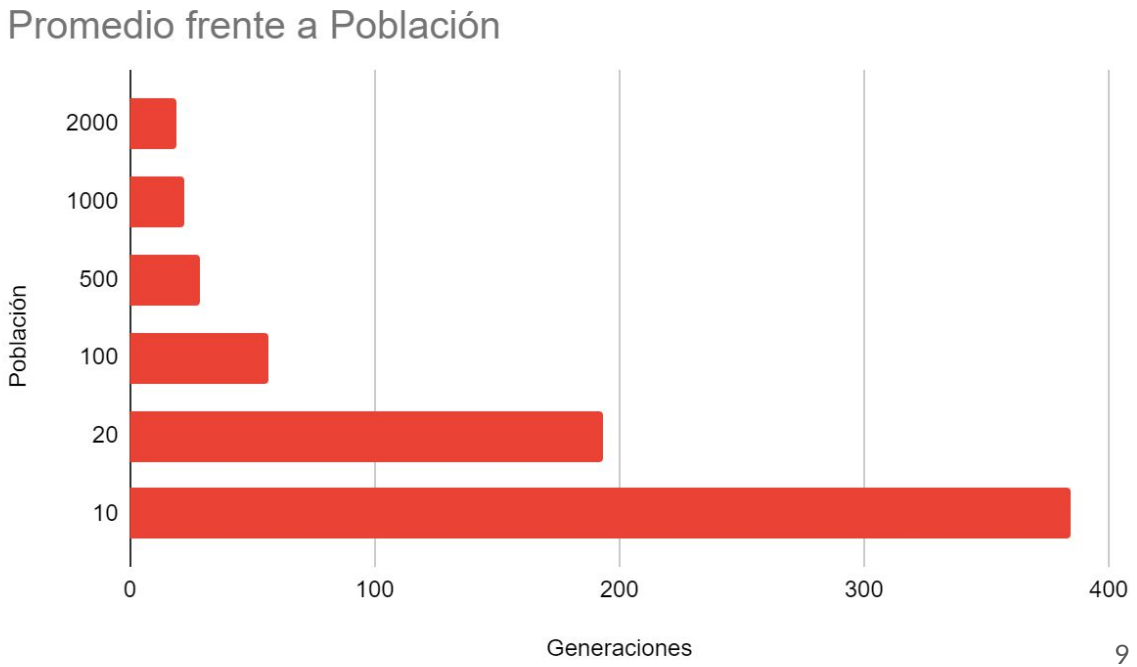


Población	Tiempo
2000	594±115
1000	253±15
500	172±9
100	100±5
20	84±3
10	83±4

Consideraciones

Elección del P

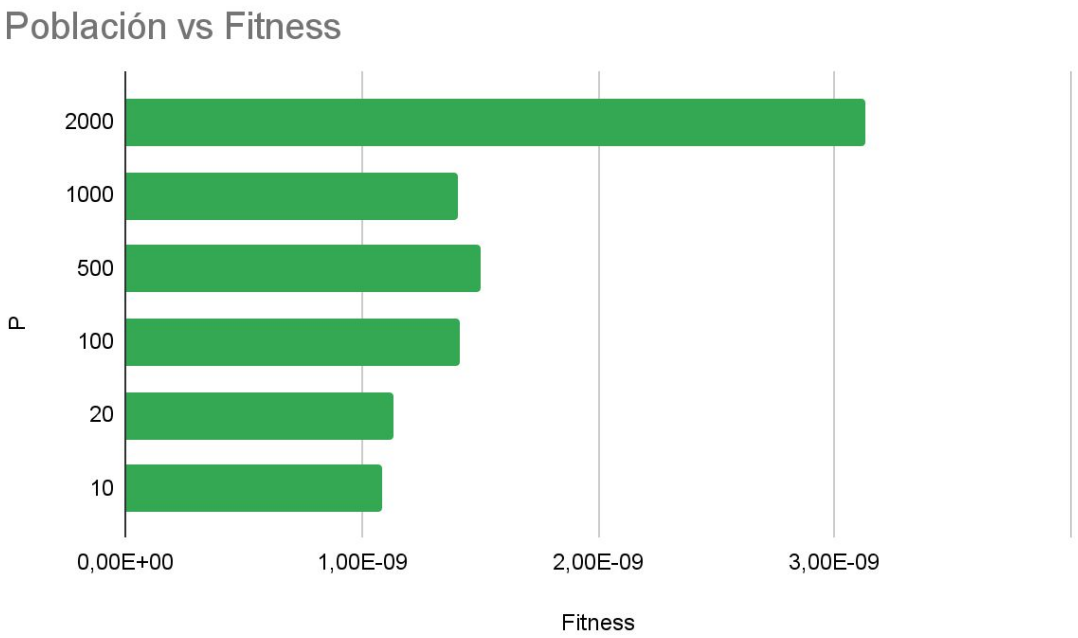
Población	Generaciones
2000	19±4
1000	22±3
500	29±3
100	56±5
20	194±4
10	384±78



Consideraciones

Elección del P

Población	Fitness
2000	(3.1±2.3)e-9
1000	(1.4±3.8)e-9
500	(1.5±2.5)e-9
100	(1.4±3.5)e-9
20	1,10E-09
10	1.1e-9



Consideraciones

Elección del P

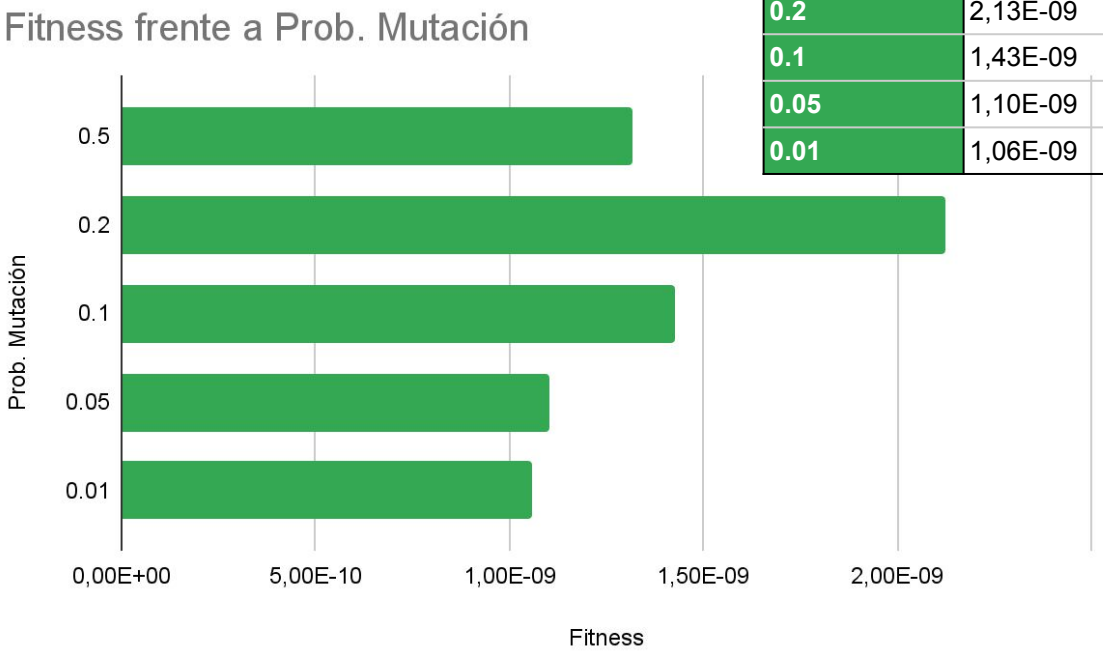
Teniendo en cuenta un término medio entre tiempo de ejecución, cantidad de generaciones creadas y fitness final se decidió trabajar con P de 500

Consideraciones

Elección del p

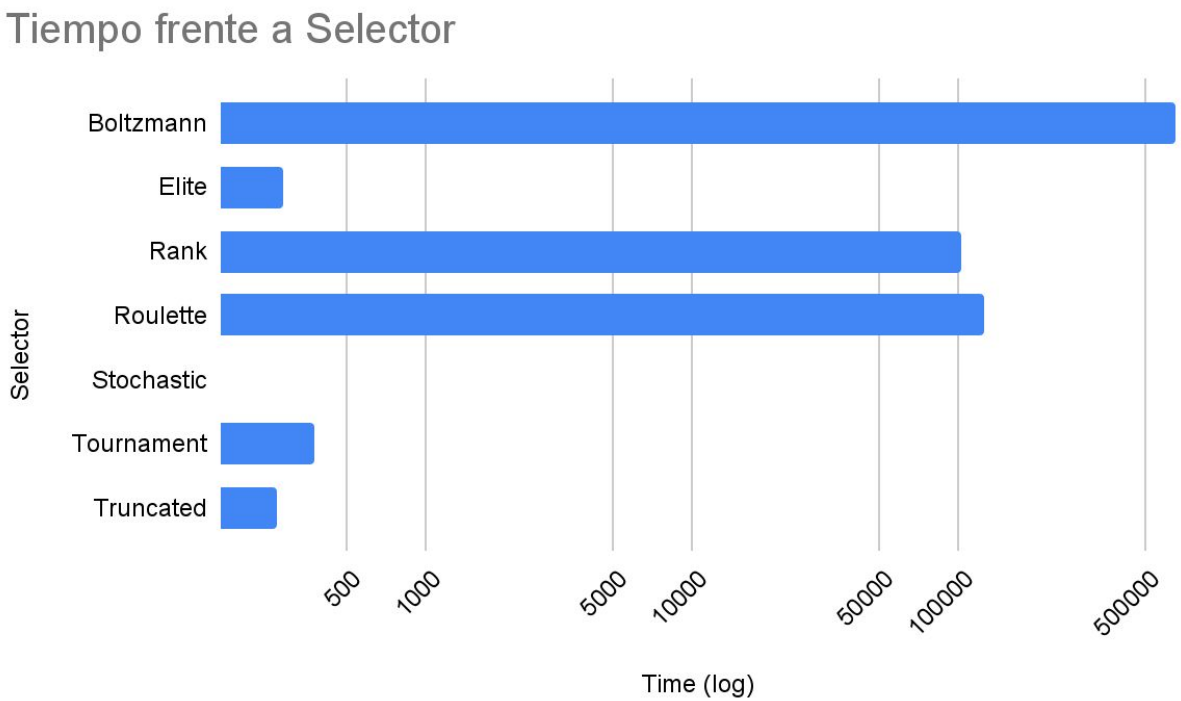
Se decidió trabajar con un p= 0.1 para evitar una mutación alta que mute constantemente retrasando la convergencia, y una baja que no permita mutaciones

Prob. Mutación	Fitness
0.5	1,32E-09
0.2	2,13E-09
0.1	1,43E-09
0.05	1,10E-09
0.01	1,06E-09



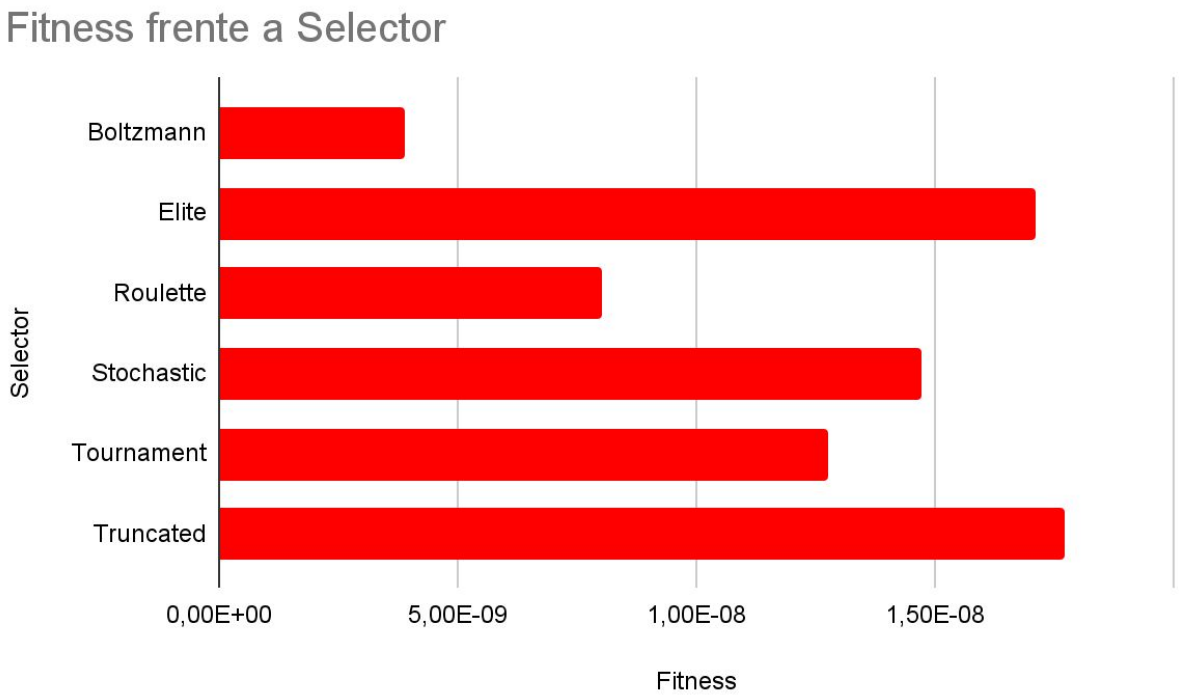
Resultados y conclusiones

Cruza: simple



Selector	Tiempo (ms)
Boltzmann	656880±15179
Elite	291±34
Rank	102521±1088
Roulette	124487±153
Stochastic	167±6
Tournament	382±12
Truncated	272±12

Cruza: simple



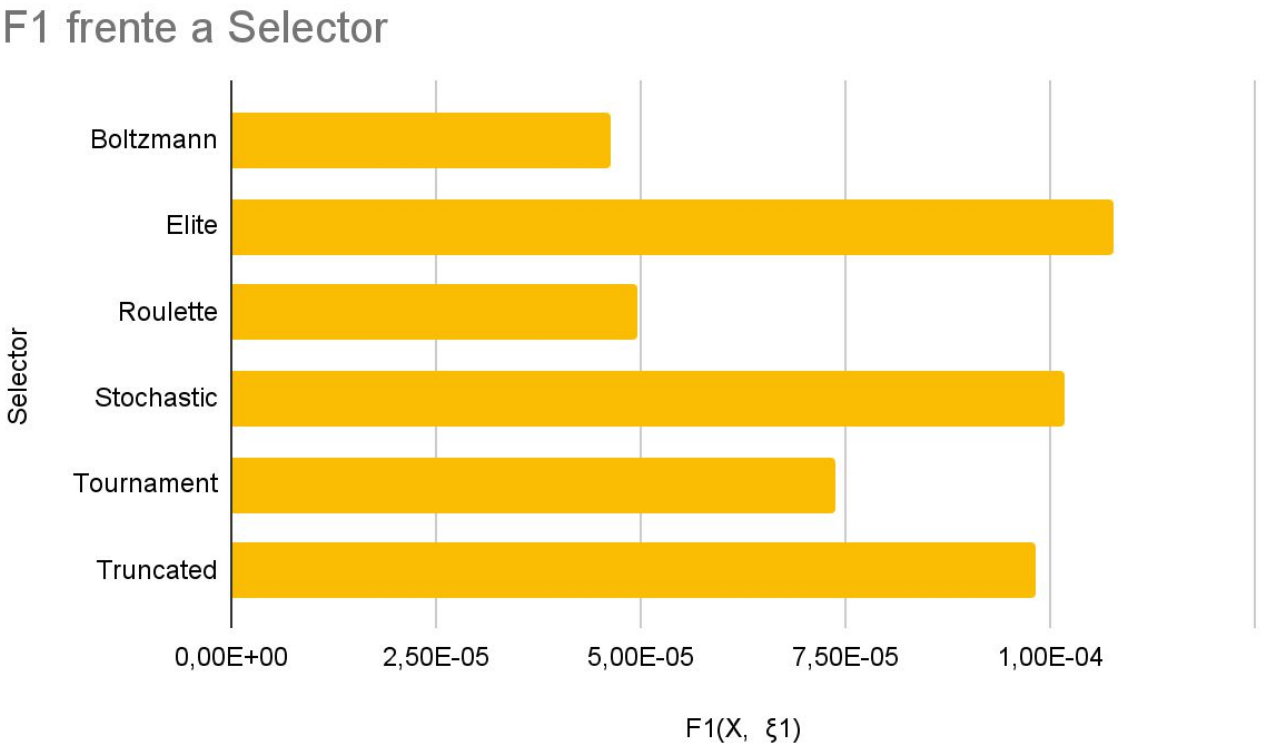
Selector	Fitness
Boltzmann	(3.9e±0.9)e-9
Elite	(10.71±3.80)e-9
Rank	0.27±0.03
Roulette	(8±2)e-9
Stochastic	(14.7±4.36)e-9
Tournament	(12.8±2.7)e-9
Truncated	(17.7±4.6)e-9

Obs: no se grafica Rank

Cruza: simple

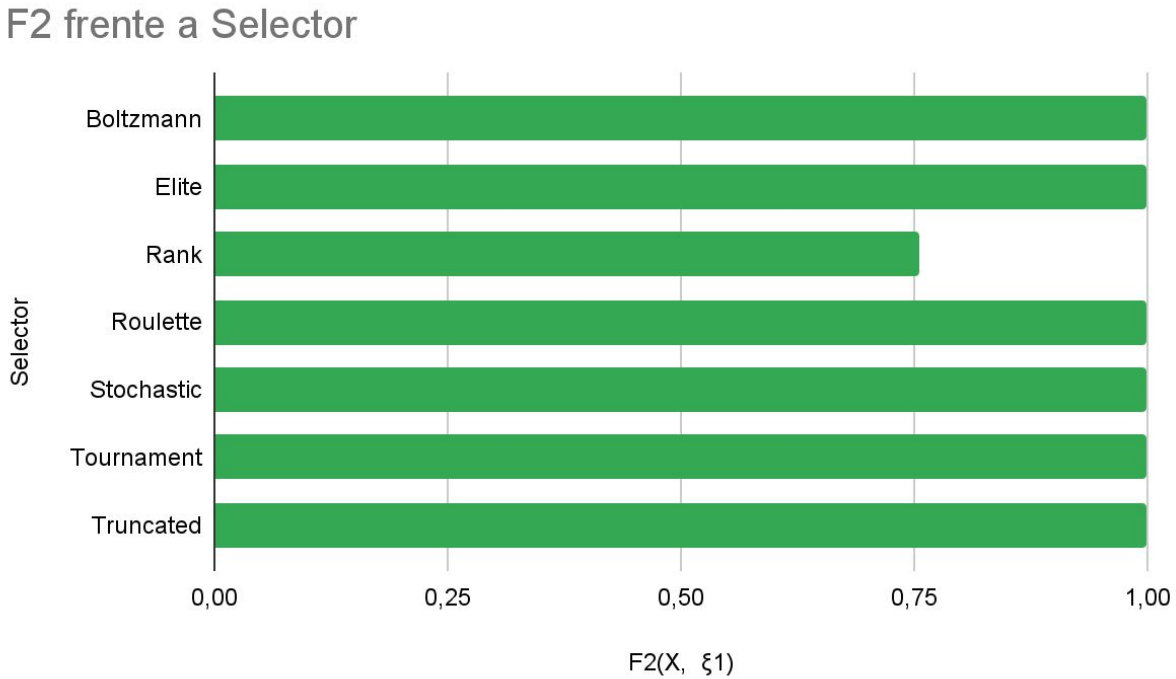
Selector	F1
Boltzmann	(4.6±0.3)e-5
Elite	(1.08±0.12)e-4
Rank	0.38±0.02
Roulette	(5±1)e-5
Stochastic	(10.18±0.31)e-5
Tournament	(7.38±0.68)e-5
Truncated	(9.82±1.21)e-5

Obs: no se grafica Rank



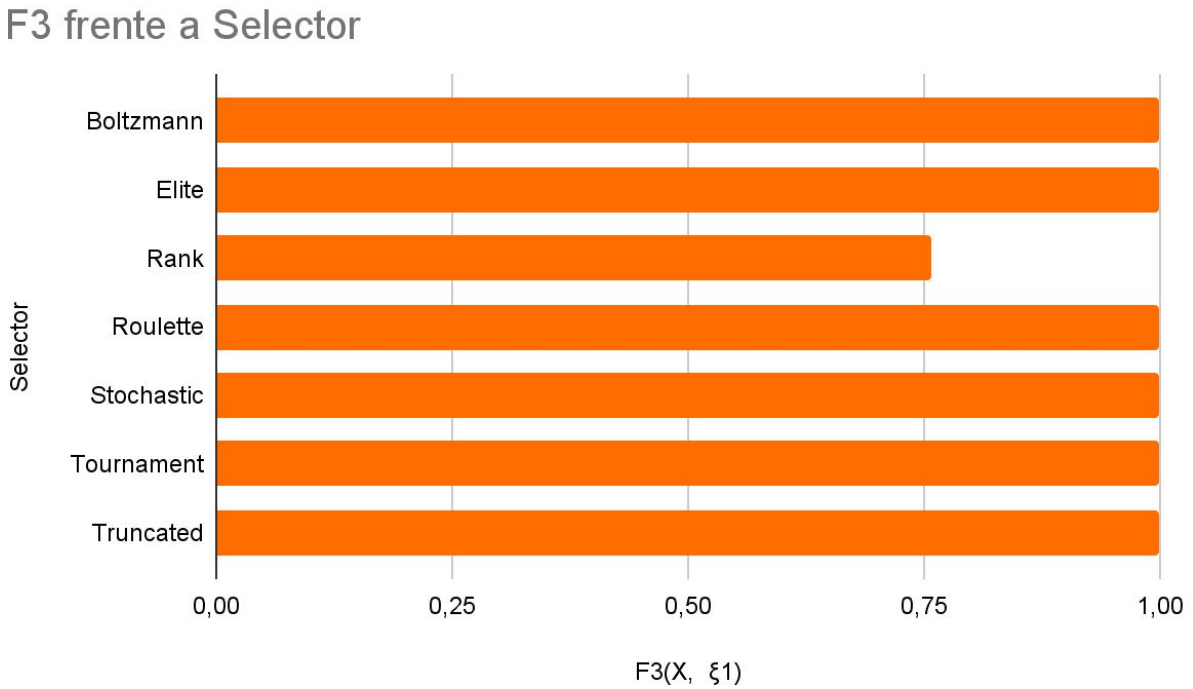
Cruza: simple

Selector	F2
Boltzmann	0.999970±0.000007
Elite	1±0.00001
Rank	0.75±0.05
Roulette	0.99±0.01
Stochastic	1±0.00001
Tournament	0.99994±0.00001
Truncated	0.99994±0.00001

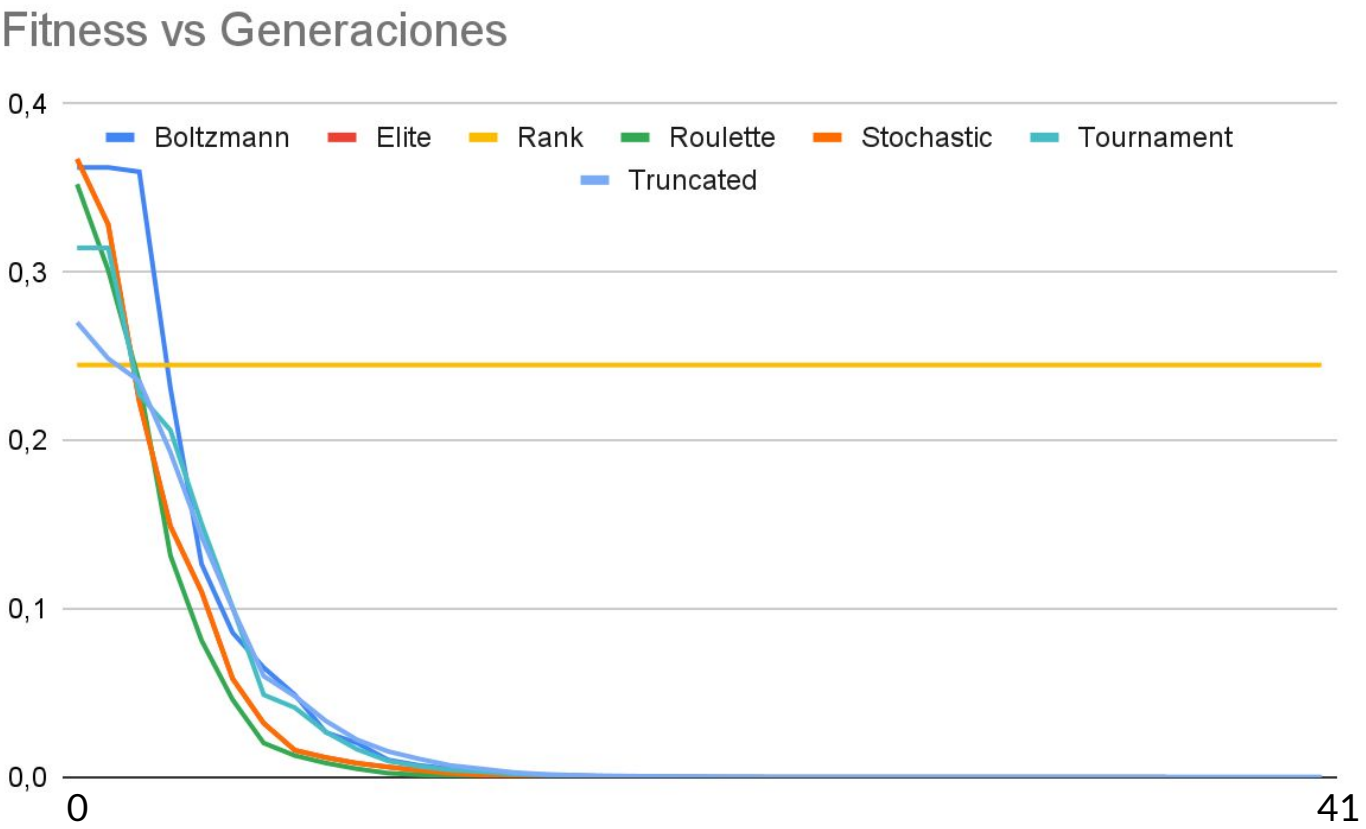


Cruza: simple

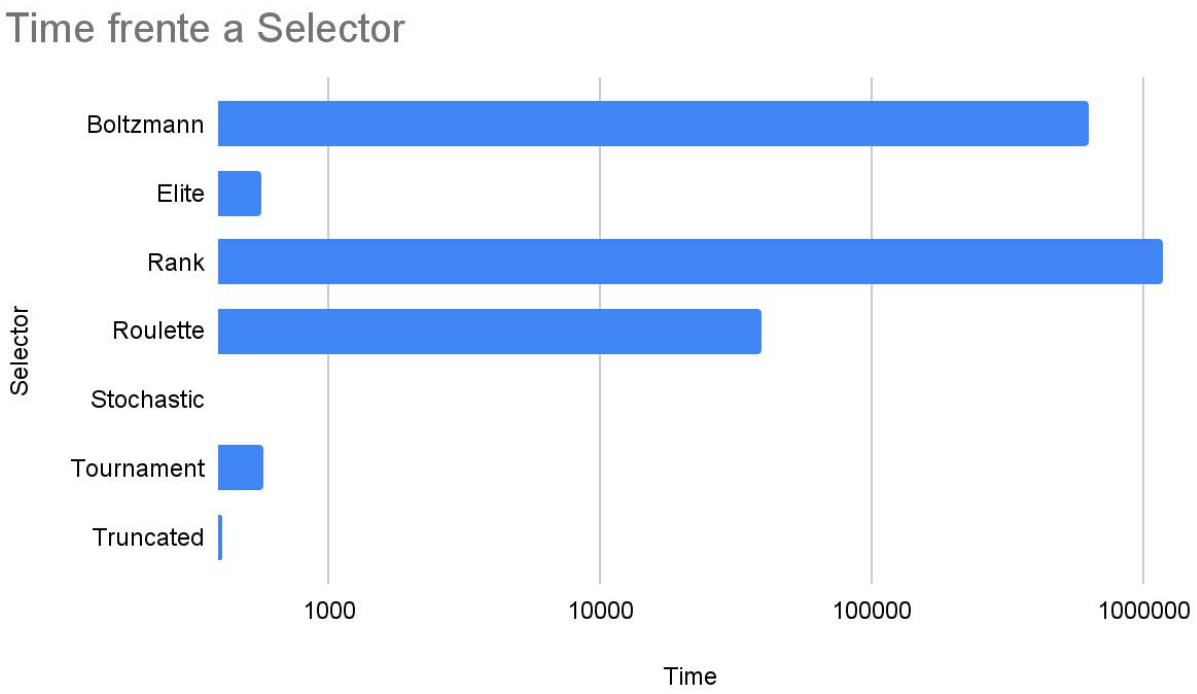
Selector	F3
Boltzmann	0.999970±0.000007
Elite	0.99995±0.00001
Rank	0.76±0.05
Roulette	0.99995±0.00001
Stochastic	0.99995±0.00001
Tournament	0.99994±0.00001
Truncated	0.99994±0.00001



Cruza: simple

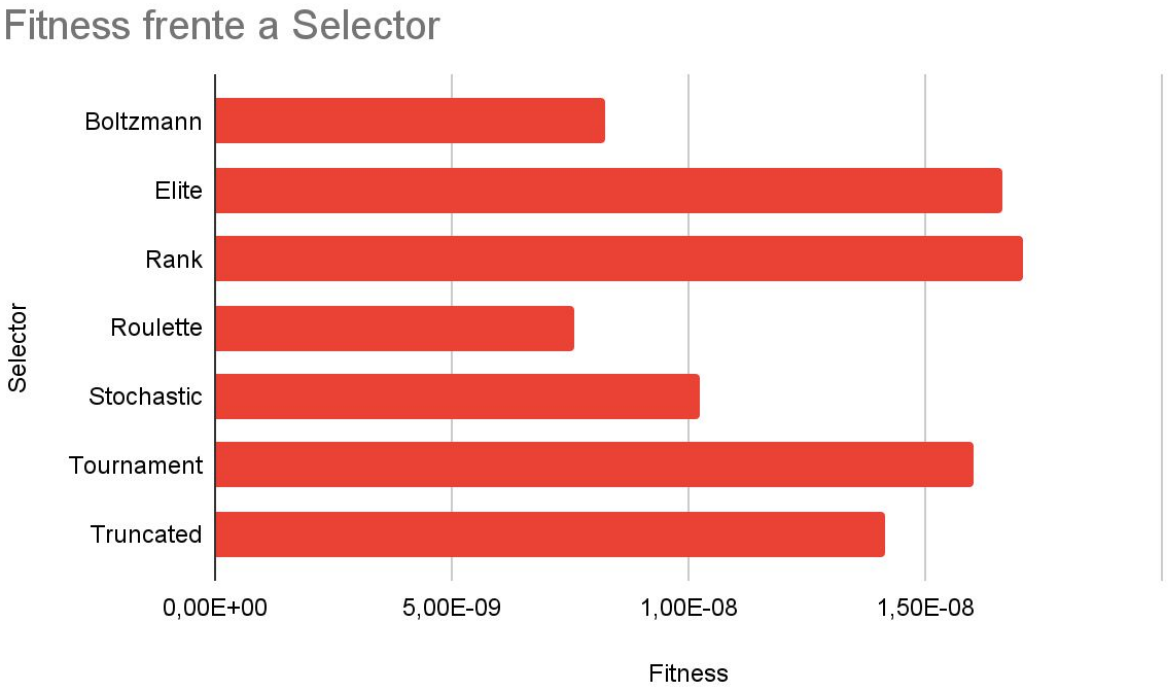


Cruza: uniforme



Selector	Tiempo (ms)
Boltzmann	632193±2922
Elite	562.5±163.3
Rank	1186648±198145
Roulette	38870±1544
Stochastic	387±27
Tournament	574±1
Truncated	405.5±198.7

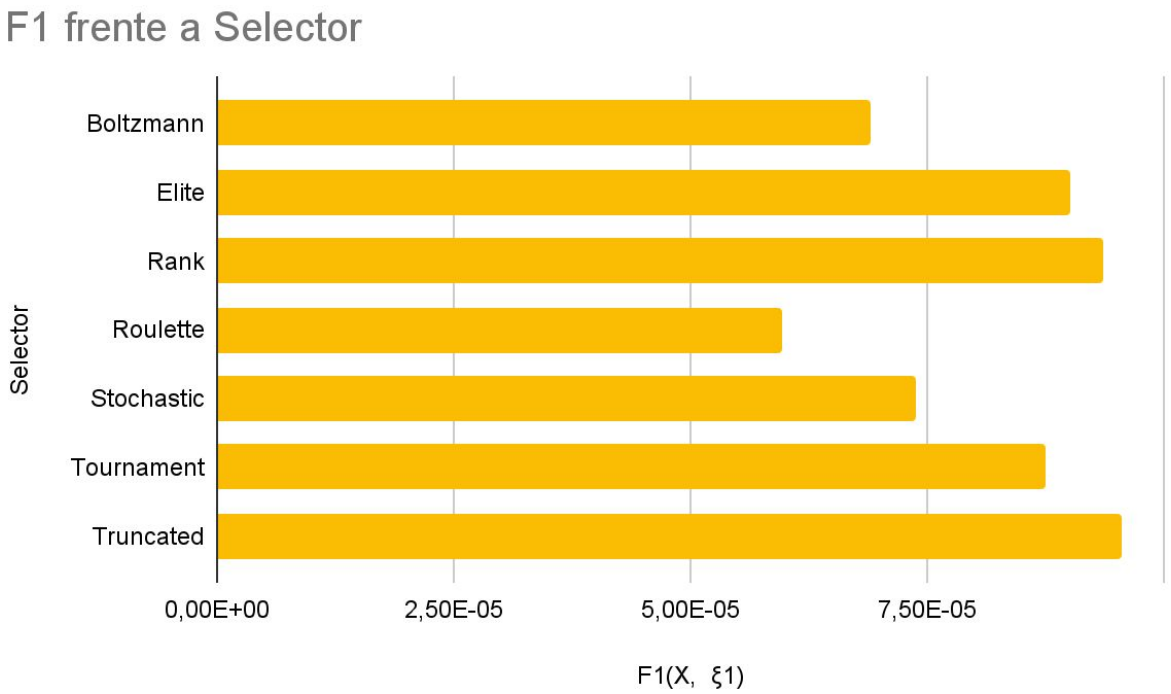
Cruza: uniforme



Selector	Fitness
Boltzmann	(8.2±0.7)e-9
Elite	(1660±3)e-11
Rank	(171e±9)e-10
Roulette	(7.6±1.2)e-9
Stochastic	(1030±2)e-11
Tournament	(160±6)e-10
Truncated	(142±4)e-10

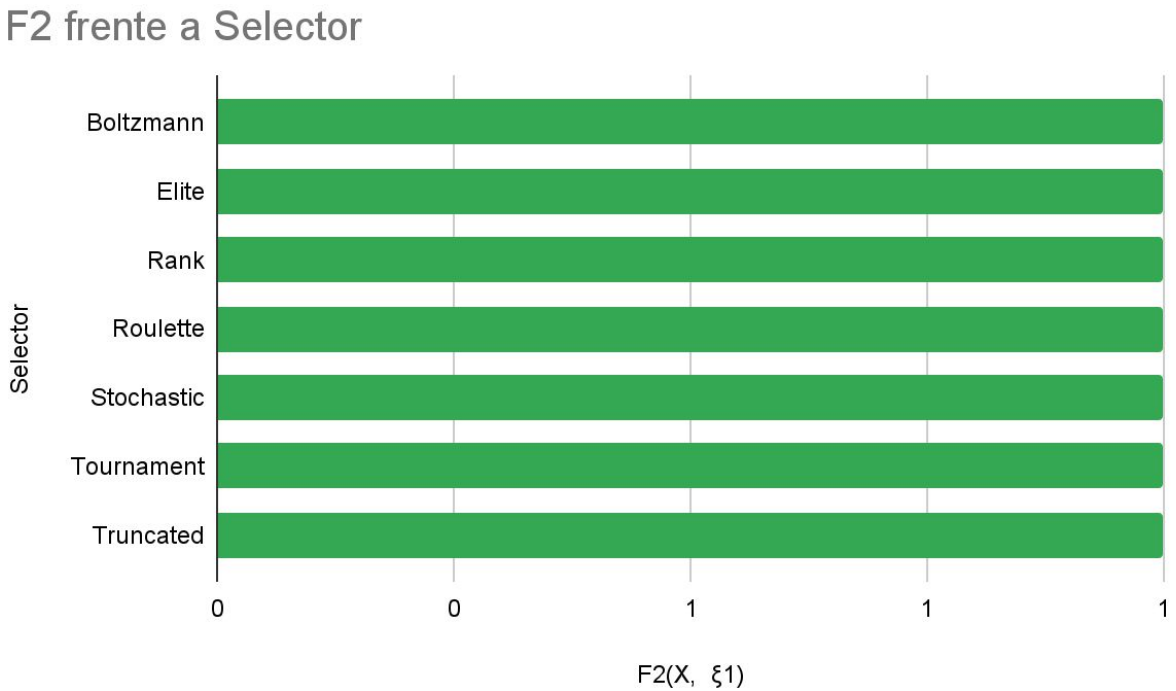
Cruza: uniforme

Selector	F1
Boltzmann	6.9e-5
Elite	9,00E-05
Rank	9,00E-05
Roulette	6.00e-5
Stochastic	7.39e-5
Tournament	8.75e-5
Truncated	9.55e-5



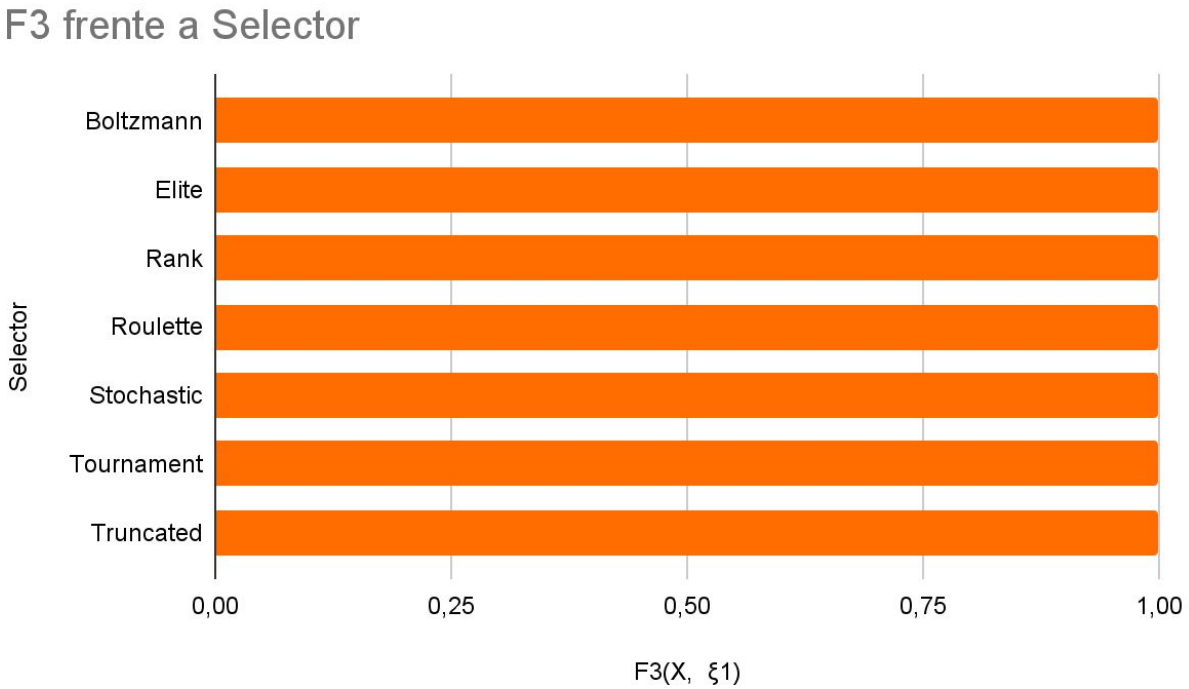
Cruza: uniforme

Selector	F2
Boltzmann	0,99996
Elite	0,99994
Rank	0.99994
Roulette	0.99996
Stochastic	0.99995
Tournament	0.99994
Truncated	0.99995

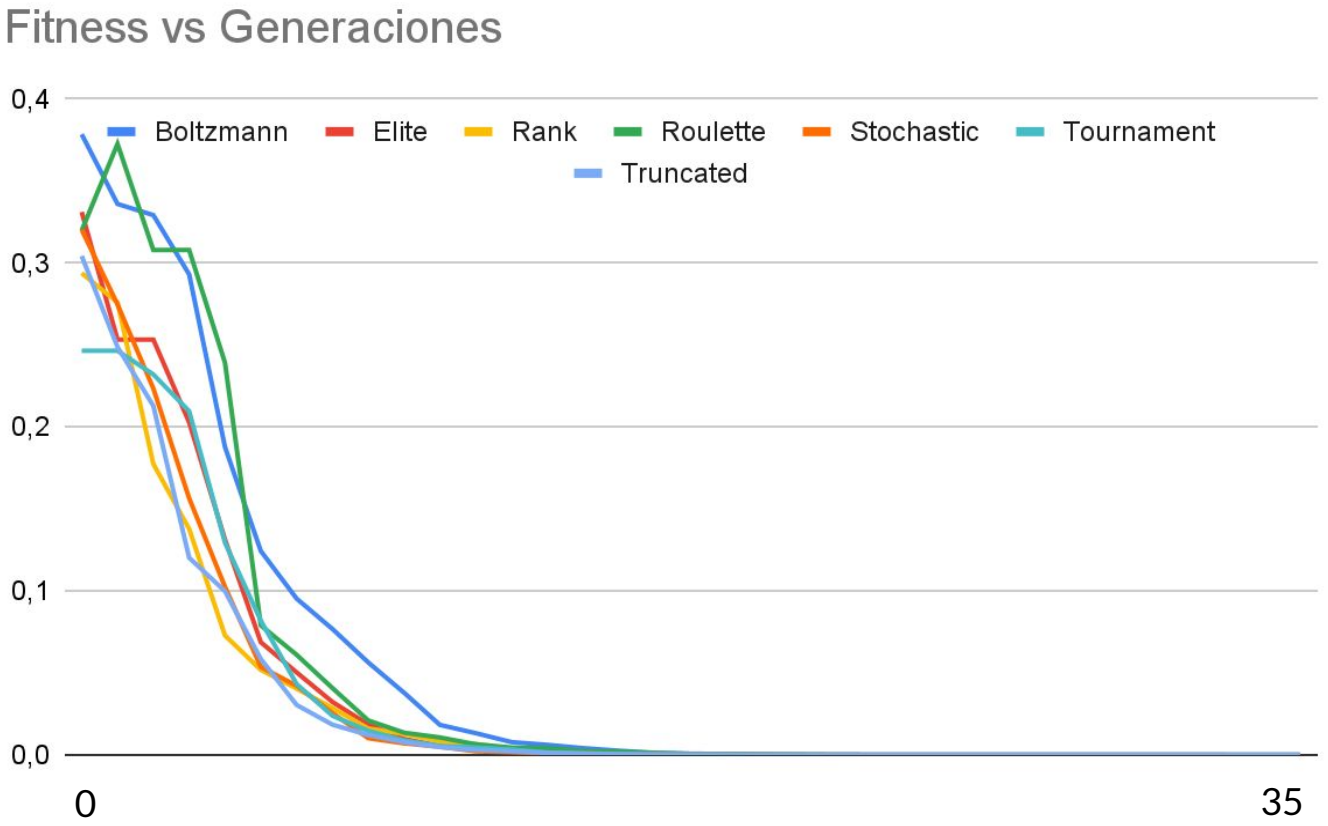


Cruza: uniforme

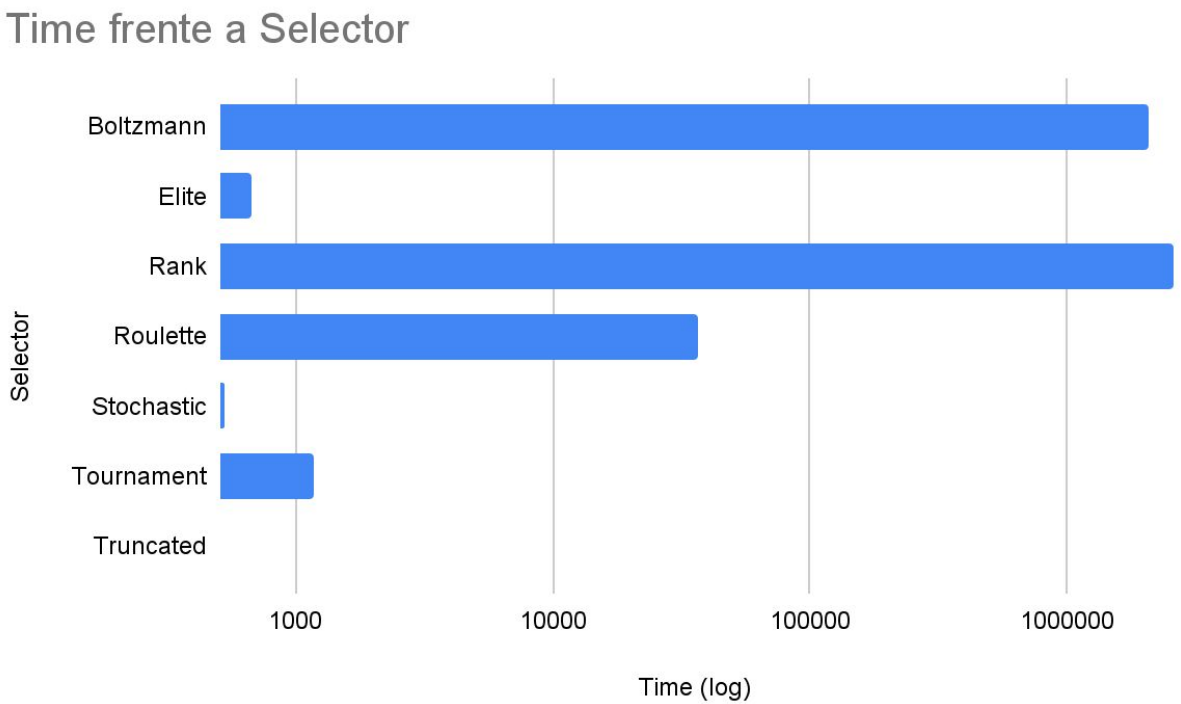
Selector	F3
Boltzmann	0.99996
Elite	0.99994
Rank	0.9994
Roulette	0.99996
Stochastic	0.99995
Tournament	0.99994
Truncated	0.99995



Cruza: uniforme

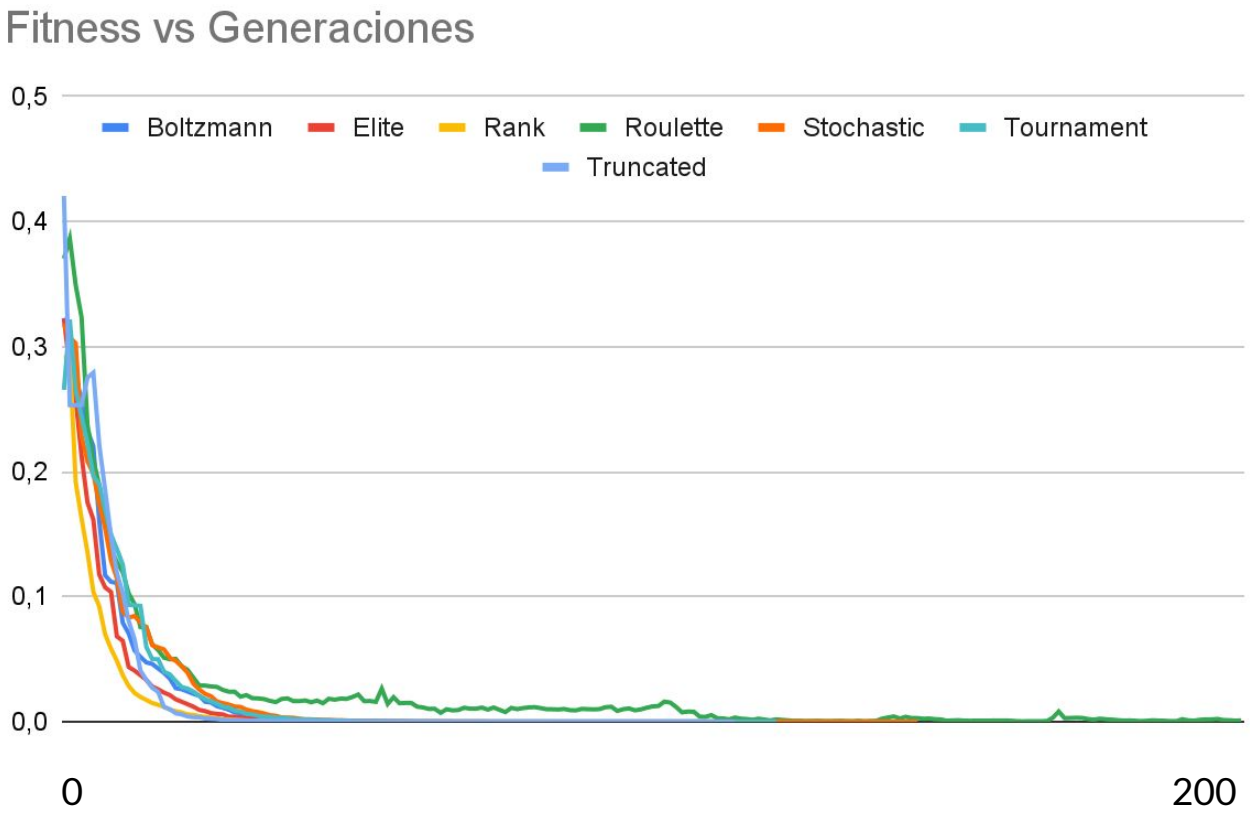


Cruza: múltiple (k=3)

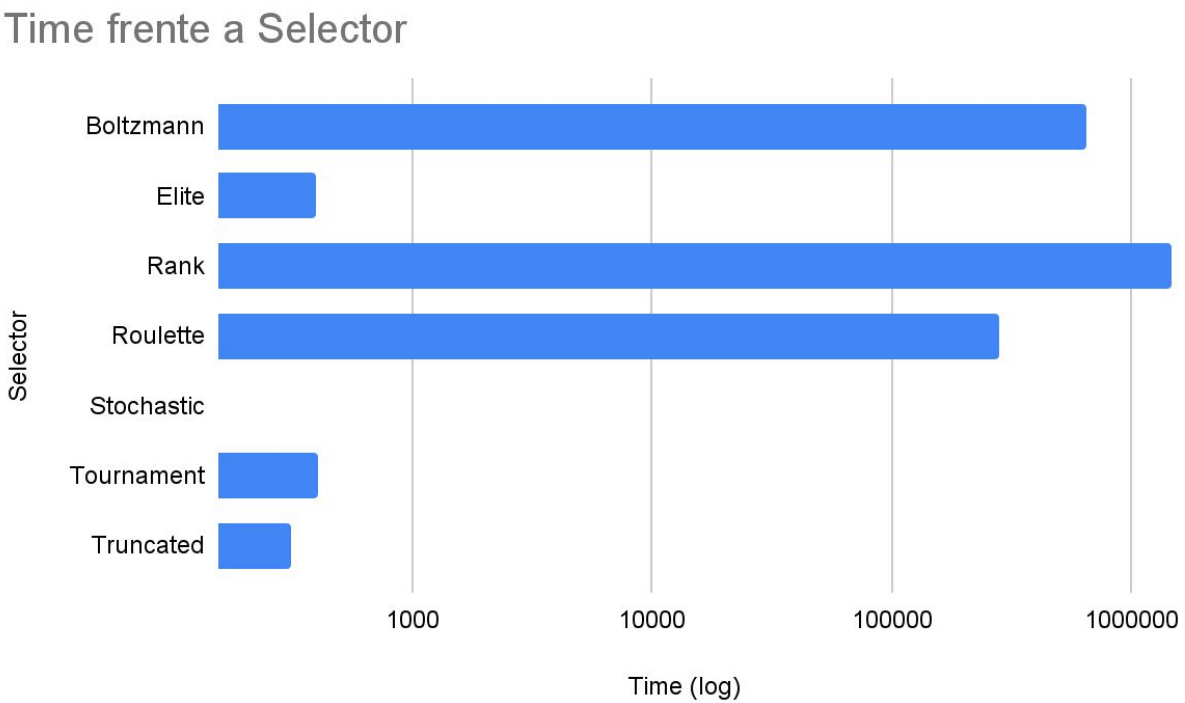


Selector	Tiempo (ms)
Boltzmann	2079905
Elite	681
Rank	2610973
Roulette	36900
Stochastic	532
Tournament	1187
Truncated	507

Cruza: múltiple (k=3)

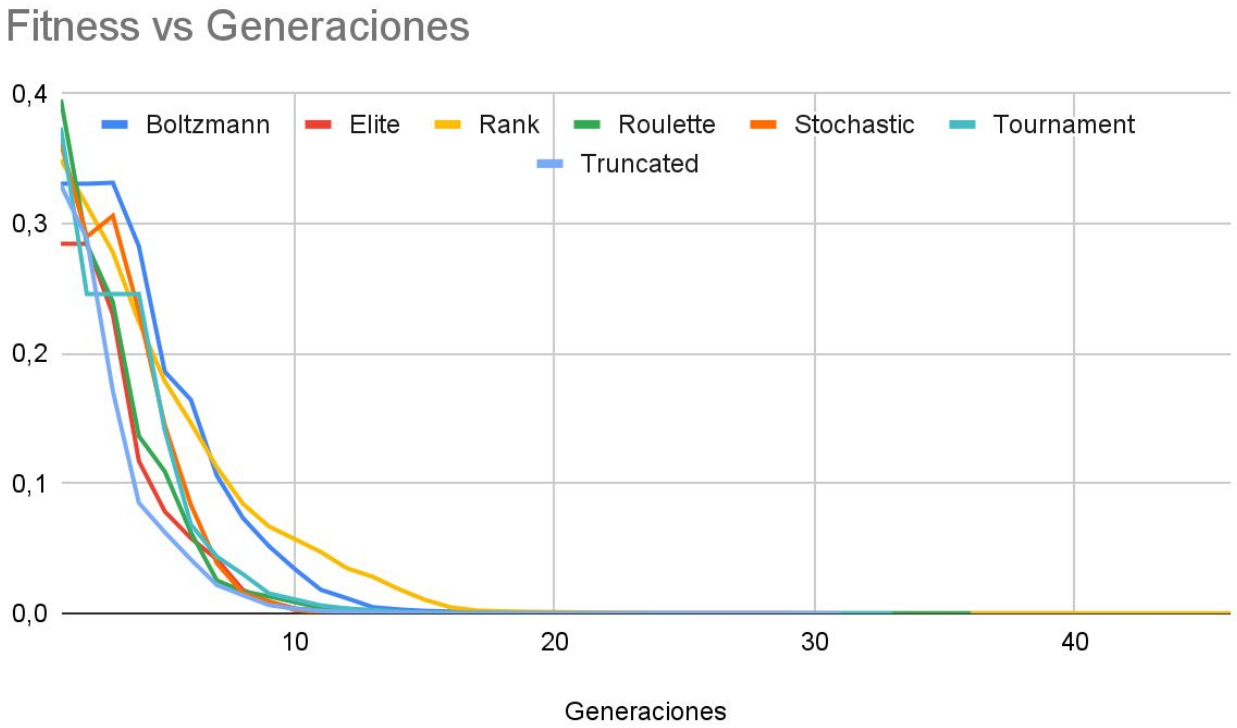


Cruza: múltiple (k=7)



Selector	Tiempo (ms)
Boltzmann	646834
Elite	395
Rank	1475768
Roulette	279465
Stochastic	154
Tournament	408
Truncated	313

Cruza: múltiple (k=7)



Conclusiones

Conclusiones

- Peores rendimientos de tiempo: Boltzman, Rank y Roulette
- Peor fitness: Rank y Roulette
- Peor F: Rank y Roulette
- Aunque Rank y Roulette tengan malos resultados, siguen siendo dentro de parámetros aceptables que podrían llevar a una solución correcta

¿Preguntas?

Muchas gracias

Lucas Catolino
Matias Ricarte