Algoritmos Genéticos

Sistemas de Inteligencia Artificial - ITBA

Carlos Sessa Lucas Pizzagalli Nicolás Purita

1. CONSIDERACIONES GENERALES

1.1 Cálculo del Fitness

El fitness del individuo es obtenido mediante una simple división del *Error cuadrático medio* que se obtiene de la red al evaluar los puntos. La fórmula para el mismo es:

$$\textbf{Fitness} = \frac{1}{ECM} \tag{1}$$

donde ECM es el Error Cuadrático Medio.

1.2 Selección de valores para Boltzman

Para elegir los valores iniciales de las temperaturas que se utilizan en el algoritmo de Boltzman (Temperatura mínima, máxima y decremento por generación) se eligieron ciertos valores que cumplan con el siguiente criterio. Al inicio de la evolución todos los individuos deben tener una probabilidad similar de ser seleccionados entre toda la población y a medida que avanzan las generaciones los individuos con mayor *fitness* deben tener mayor probabilidad de ser seleccionados. Los valores que se eligieron de Boltzman para hacer las pruebas son variables dependiendo de la cantidad de iteraciones que se desea realizar, y suponiendo un *fitness* en un rango determinado. Esto logra una gran diversidad en un comienzo, y luego va refinando la población descartando de a poco los individuos menos aptos.

2. RESULTADOS

Con el fin de obtener resultados comparables entre sí se define un contexto base para todas las pruebas, con la intención de obtener distintos resultados se realizan cambios sobre esta configuración:

- **Cantidad de individuos:** 52.
- **Mutación:** Clásica. Con una probabilidad de mutar un individuo de 0.5 y de mutar un locus de 0.03.
- **Cruce:** Clásico.
- **Backpropagation:** 0 (Desactivado)
- **Selección:** Elite+Rulette (Seleccionando 5 para Elite y 23 para Rulette).
- **Reemplazo:** Elite+Rulette (Seleccionando 6 para Elite y 18 para Rulette).
- **Criterios de corte:** Máxima 1000 generaciones o Contenido, si en 60 generaciones no mejora un 1% el mejor individuo corta.

2.1 Boltzman

Con el objetivo de analizar el desempeño de este método de selección y reemplazo, se decidió probar con una configuración Boltzman-Boltzman (donde la selección y el reemplazo se realiza con dicho método) con una configuración como la siguiente:

1. *Temperatura inicial*: 40
2. *Temperatura mínima*: 0.4
3. *Decremento por generación*: 0.05

Esto quiere decir que la temperatura mínima se alcanza en 800 generaciones aproximadamente, o sea que entre la primera y la generación número 800, se van seleccionando cada vez más proporción de individuos con *fitness* más alto. Como se puede observar en la figura 1, tanto el valor medio como el mejor *fitness* tienden a empeorar, comportarse de una forma bastante errática y luego comenzar a mejorar de a poco. Esto se debe a que, como se ha descripto anteriormente, en un comienzo se tiende a elegir de una manera casi totalmente aleatoria, pero a medida que se acerca y supera la generación 600 y aun más claramente cuando se alcanza la temperatura mínima, la selección se realiza más a conciencia, dándole prioridad a los individuos con más aptitud.

2.2 Mutación No uniforme

Con el objetivo de comparar el comportamiento del sistema según como varía el decaimiento de la probabilidad de mutación a medida que las generaciones avanzan. Por lo tanto, se decidió comparar mutaciones no uniformes con decaimientos de 5%, 10% y 15% cada 30 generaciones. Los resultados de las mismas se pueden observar en las figuras 4(a) , 4(b) y 4(c) respectivamente. De las mismas se puede observar a simple vista, como en todos los casos termina por contexto (60 generaciones con un cambio menor a 1%), sin embargo, se puede ver como al ser el decremento mayor, el algoritmo termina antes. Esto se debe a que al haber muchas menos mutaciones, no hay introducción de elementos nuevos y no se llega a cambiar lo suficiente las redes como para obtener nuevas combinaciones y se consigan mejores resultados. Este estancamiento, también se evidencia en el menor desvío estándar.

2.3 Crossover

Se realizaron varias corridas distintas en donde se modifica unicamente el método de *cruce*. En la figura 5 se pueden observar los 4 métodos distintos. Se observa en esta comparación que el mejor resultado obtenido es con el método de

cruce *Uniforme* pero esto no permite sacar una conclusión definitiva.
Lo que si aporta esto es que el método de cruce aporta diversidad a la población para poder explorar mayor espacio de individuos.

2.4 Selección y Reemplazo

En esta sección se obtuvieron varios resultados donde se pueden sacar algunas conclusiones acerca de los métodos de selección y reemplazo.
En primer lugar el comportamiento del algoritmo utilizando como método de Selección y Reemplazo **Ruleta**, se puede observar como el *fitness* tiende a oscilar. Este comportamiento es de esperar ya que este método no garantiza el mejor individuo para la próxima generación sino que tiene más probabilidad de ser seleccionado. En la figura 3 se observa este comportamiento explicado.
Otro punto a destacar es que si utilizamos el método *Elite* (como selección y reemplazo) no aporta diversidad a la población, dado que siempre selecciona los **N** mejores individuos para la siguiente generación. Por lo tanto la utilización de *Elite* con *Boltzmann* o *Rulette* se obtienen buenos resultados. Esta mejora se produce ya que estos dos métodos aportan diversidad a la población. Estas comparaciones se pueden observar en las Figuras 2(b), 2(c) y 2(a).

2.5 Backpropagation vs Algoritmos Genéticos con Backpropagation

En las figuras 6(a) y 6(b) se puede observar como mejora la *Red neuronal* utilizando Algoritmos Genéticos. El primer punto a destacar es el comportamiento de la curva del *Fitness*, en la figura 6(b) se puede observar que se obtiene un mejor resultado que la figura 6(a) y además no posee esas oscilaciones que presenta la red al utilizar método Batch como corrección. Se puede observar que los dos algoritmos mejoran a traves del tiempo, sólo que en la figura 6(b) lo hace con mayor velocidad.

3. CONCLUSIONES

Se puede concluir que el algoritmo genético puede resolver el problema, sin la utilización del operador *Backpropagation*. En cambio, sin embargo si el operador Backpropagation toma juego en el desarrollo del Algoritmo Genético, los resultados obtenidos son mejores con un mismo o menor costo computacional.

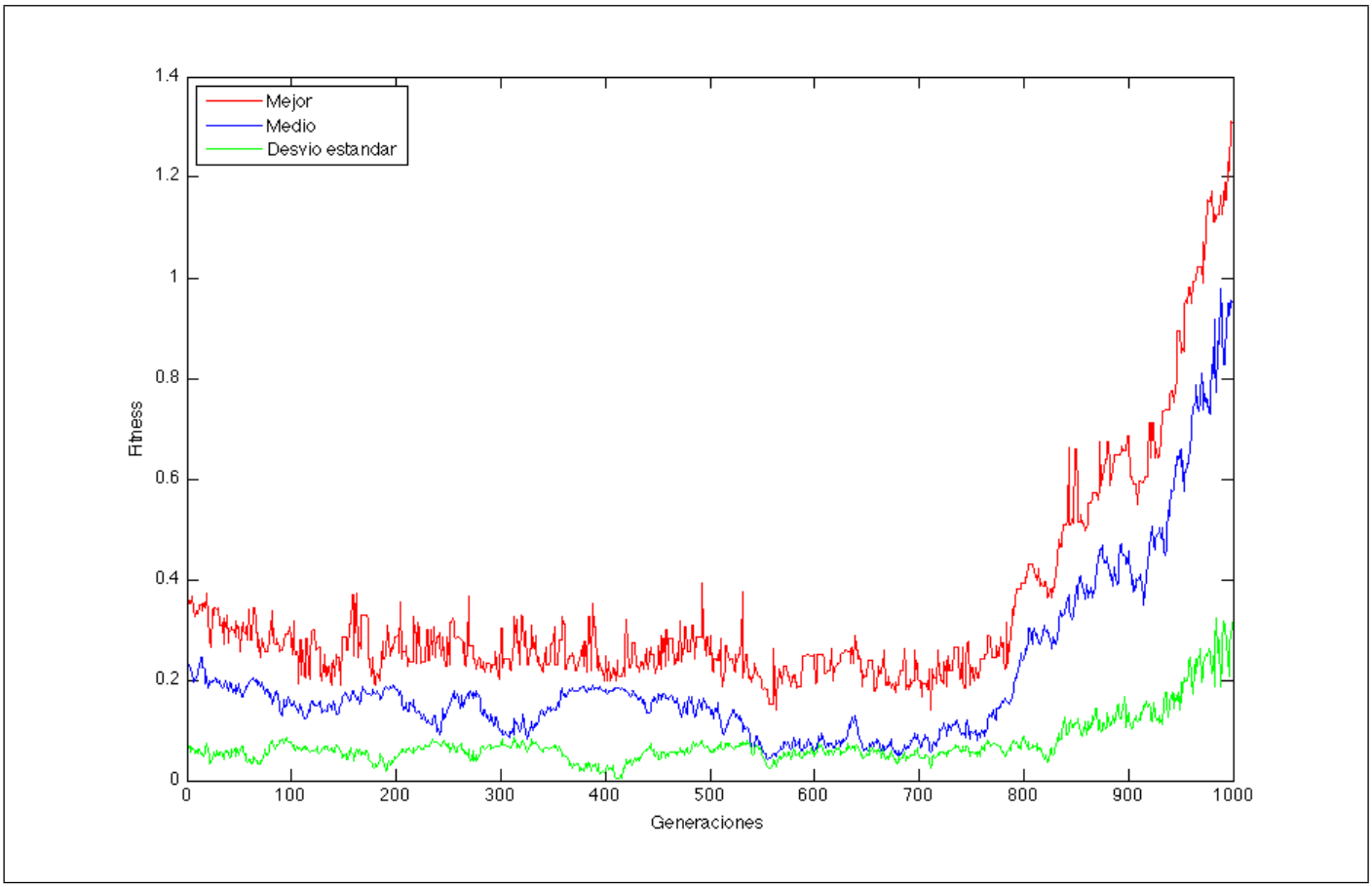
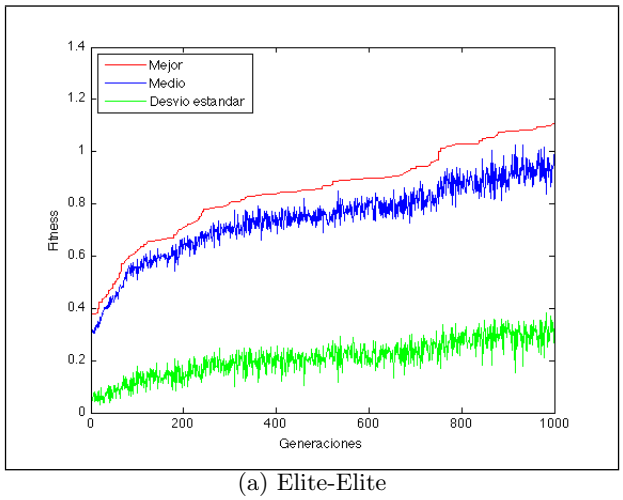
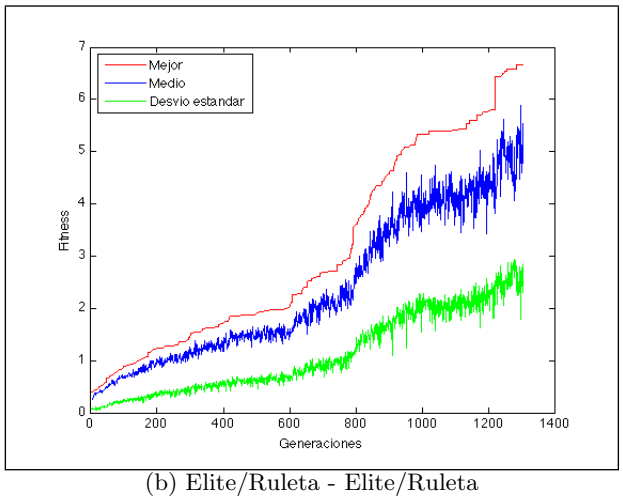


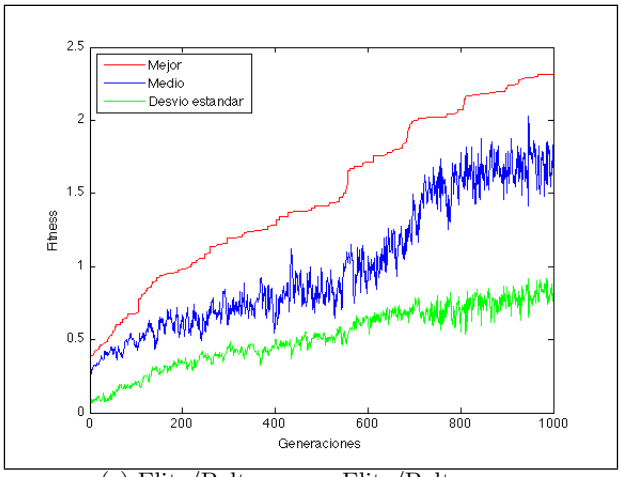
Figura 1: Selección y reemplazo por método de Boltzman



(a) Elite-Elite



(b) Elite/Ruleta - Elite/Ruleta



(c) Elite/Boltzmann - Elite/Boltzmann

Figura 2: Comparación Elite, Elite/Ruleta y Elite/Boltzmann

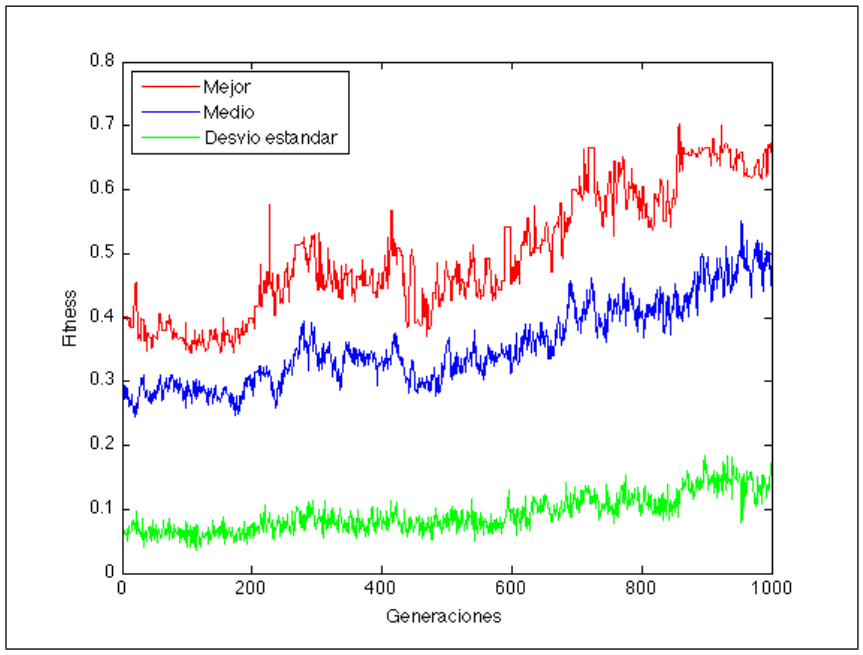
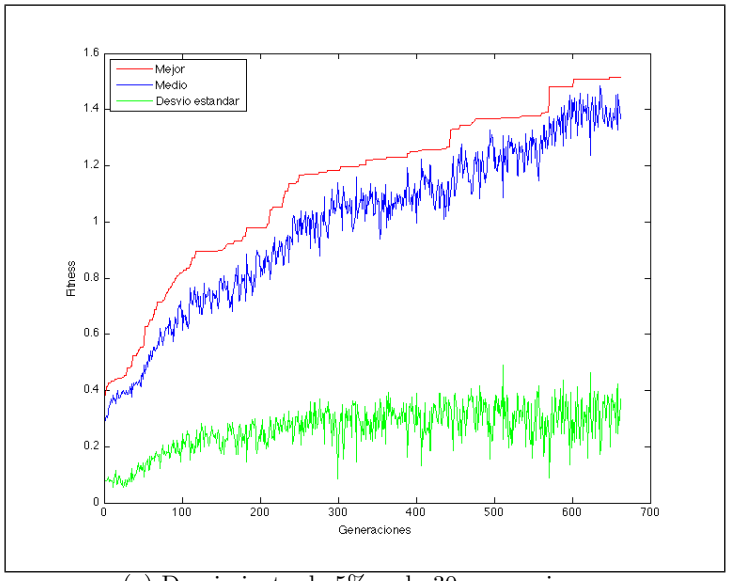
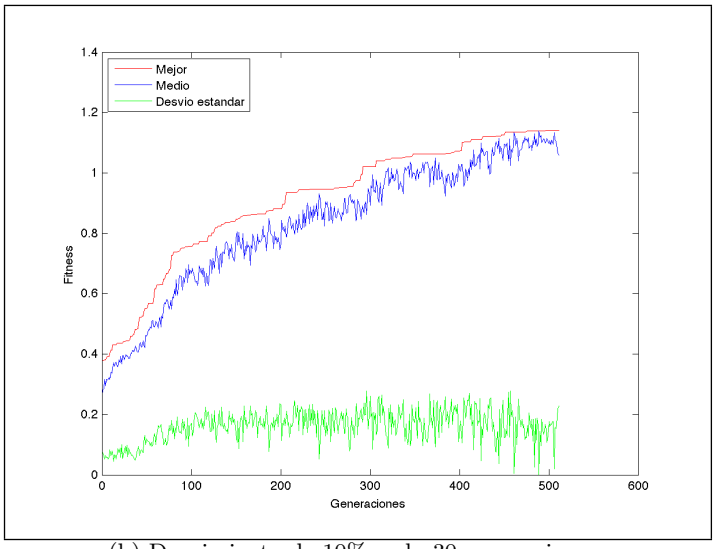


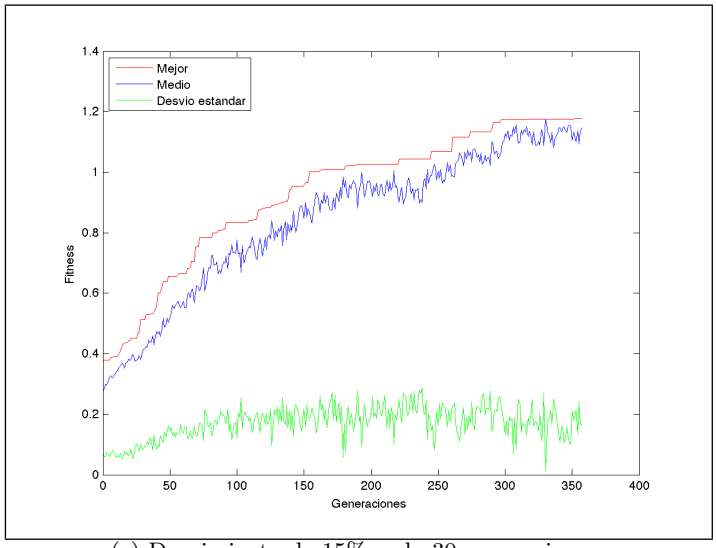
Figura 3: Selección y reemplazo por método de Ruleta



(a) Decaimiento de 5% cada 30 generaciones

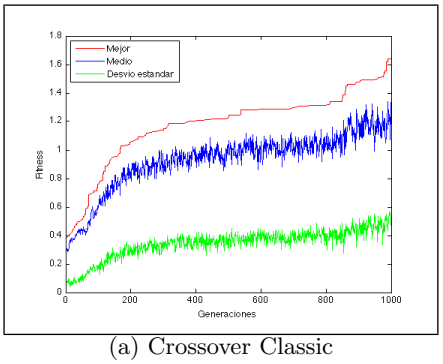


(b) Decaimiento de 10% cada 30 generaciones

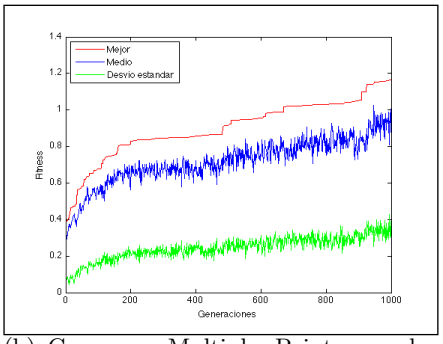


(c) Decaimiento de 15% cada 30 generaciones

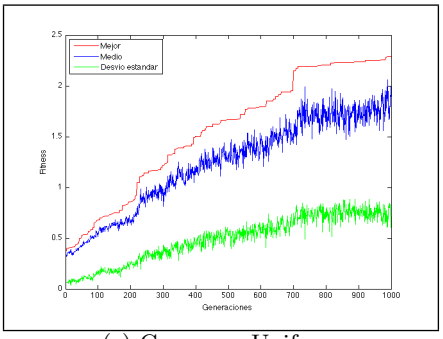
Figura 4: Influencia del decaimiento en mutaciones no uniformes



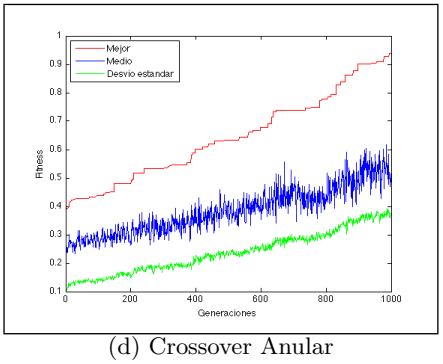
(a) Crossover Classic



(b) Crossover Multiple Point con dos puntos

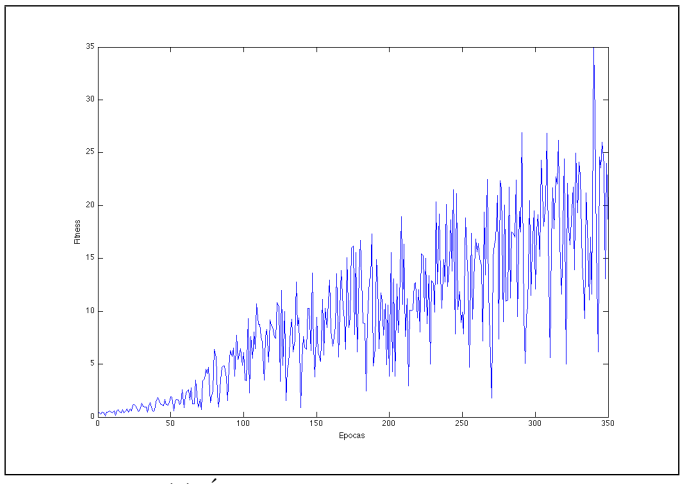


(c) Crossover Uniform

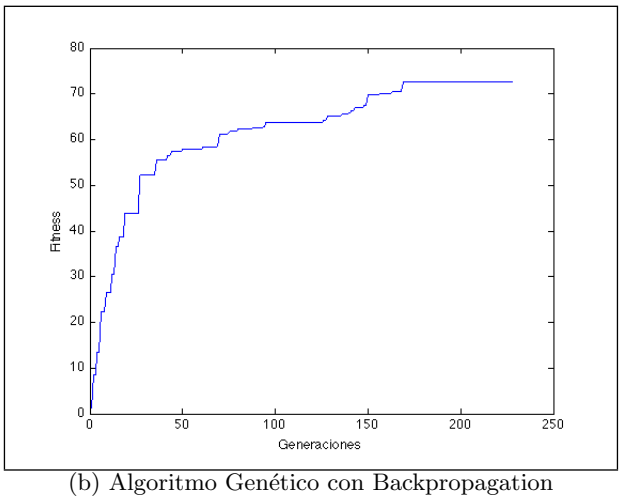


(d) Crossover Anular

Figura 5: Comparación entre distintos métodos de crossover



(a) Únicamente Backpropagation



(b) Algoritmo Genético con Backpropagation

Figura 6: Comparación entre Backpropagation y Algoritmo Genéticos con Backpropagation

Listing 1: Archivo de configuración simple

```
popSize = 52
architecture = 2
generationGap = 0.5

mutation = Classic
mutationProbability = 0.5
Mutation.alleleProb = 0.03

Backpropagation.probability = 0

crossover = Classic
crossoverProbability = 1


selection = Elite / Rulette
Elite.toSelect = 5
Rulette.toSelect = 23

replacement = Elite / Rulette
replacement.Elite.toSelect = 6
replacement.Rulette.toSelect = 18

ending = MaxGeneration / Content
MaxGeneration.iterationToEnd = 500

Content.improvement = 0.01
Content.iterationToImprove = 60
Content.iterationToImprove = 15
```