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 las opciones malas.

2. RESULTADOS Y CONCLUSIONES

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 donde tenemos una probabilidad de 0.015 de mutar.
- **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 500 generaciones o Contenido, si en 60 generaciones no mejora un 1% el mejor individuo corta.

2.1 Backpropagation vs Algorítmo Genéticos con Backpropagation

En las figuras 1(a) y 1(b) se puede observar como mejora la *Red neuronal* utilizando Algorítmos Genéticos. El primer punto a destacar es el comportamiento de la curva del *Fitness*, en la figura 1(b) se puede observar que se obtiene un mejor resultado que la figura 1(a) y ademas no posee esas oscilaciones que presenta la red al utilizar método Batch como corrección. Se puede observar que los dos algorítmos mejoran a traves del tiempo, solo que en la figura 1(b) lo hace con mayor velocidad.

2.2 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:

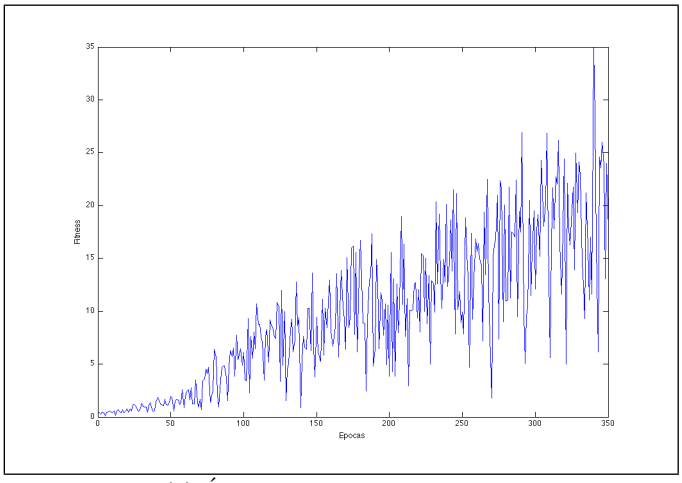
1. *Temperatura inicial*: 50
2. *Temperatura mínima*: 0.5
3. *Decremento por generación*: 0.5

Esto quiere decir que la temperatura mínima se alcanza en 100 generaciones aproximadamente, ósea que entre la primera y la generación número 100, se van seleccionando cada vez más proporción de individuos con *fitness* más alto. Como se puede observar en la figura **??**, 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 random, pero a medida que se acerca y supera la generación 100 y se alcanza la temperatura mínima, la selección se realiza más a conciencia, dandole prioridad a los individuos con buen desempeño.

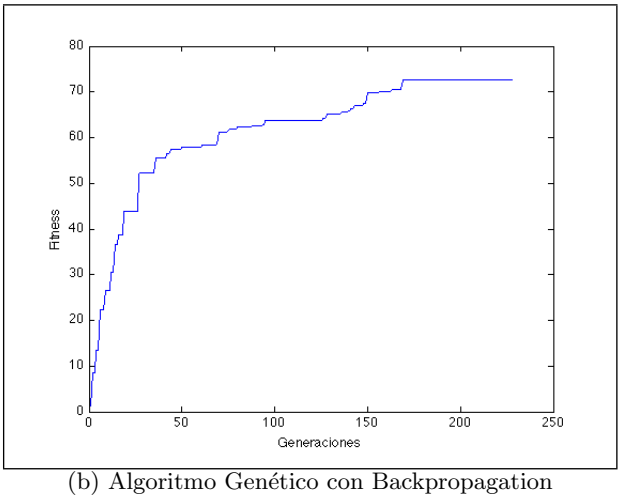
2.3 Mutación No uniforme

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 disidió comparar mutaciones no uniformes con decaimientos de 5%, 10% y 15% cada 30 generaciones. Los resultados se las mismas se pueden observar en las figuras **??** , **??** y **??** 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.



(a) Únicamente Backpropagation



(b) Algoritmo Genético con Backpropagation

Figura 1: Comparación entre Backpropagation y Algoritmo Genéticos con Backpropagation

Fitness	Generación	Corte	p_m
0.22501570790437642	60	Contenido	0.05
0.21842640558999668	60	Contenido	0.1
0.22113522454399956	67	Contenido	0.3
0.22050756611447994	72	Contenido	0.5

Tabla 1: Configuración simple variando la p_m

Fitness	Generación	Corte	p_m	decrecimiento (%)	dec. gen.
0.21222349890474348	50	Content	0.6	5	30
0.22795744299992385	77	Content	0.6	10	30
0.21740760693242447	65	Content	0.6	15	30

Tabla 2: Configuración simple método de mutación No Uniforme

Fitness	Generación	Corte	p_m	Cruce
0.22501570790437642	60	Contenido	0.05	Clásico
0.20662153685787393	50	Contenido	0.05	Gene
0.21418198155450324	50	Contenido	0.05	Multiples puntos con 2
0.21674758534799746	70	Contenido	0.05	Uniforme con 0.3 prob
0.20662153685787393	50	Contenido	0.05	Anular

Tabla 3: Configuración simple variando el método de cruce

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
```

Nombre .properties	Fitness	Generación	Corte
bp_seb_rer_cg_mc_ec.properties	16.720276062651436	69	Contenido
bp_ser_reb_mc_cc_emc.properties	11.941655508072634	58	Contenido
bp_se_re_mc_cc_emc.properties	4.825691593990156	51	Contenido
bp_seb_rer_cmp_mc_emc.properties	12.57897997517028	36	Contenido
bp_seb_rer_mc_ca_emc.properties	13.560979676949982	50	Contenido
bp_seb_rer_mnu_cc_emc.properties	12.992838764610976	36	Contenido

Tabla 4: Distintas Configuraciones con backpropagation