# Sistemas de Inteligencia Artificial

## Trabajo Práctico Especial 3

### Algoritmos Genéticos

5 de Junio de 2019

---


Grupo 9

*De Rienzo, Constanza - 56659*
*Godfrid, Juan - 56609*
*Osimani, Agustina - 57526*
*Radnic, Pablo Ignacio - 57013*


---

# Objetivo

El objetivo del trabajo práctico es implementar un motor de algoritmos genéticos para encontrar la configuración óptima de ítems de vestimenta y altura para optimizar el desempeño de un personaje en un videojuego.

# Introducción

El desempeño de un personaje se calcula a partir de la altura y cinco atributos: Fuerza, Agilidad, Pericia, Resistencia y Vida. Estos cinco atributos vienen dados por los distintos ítems de vestimenta que posee el personaje. Cada ítem de vestimenta aporta una cierta cantidad de cada uno de los atributos, y la suma de ellas determina el valor total de cada atributo para el personaje.
El cálculo del desempeño también depende de la clase del personaje, que puede ser una de las siguientes: Arquero, Defensor, Guerrero y Asesino y del número que posea, por ejemplo Defensor número 2.

# Implementación

El desarrollo de este trabajo se realizó en el lenguaje Python y se utilizaron las librerías *numpy, pandas, matplotlib, statistics y PyQt5.*
En primera instancia se analizó el problema propuesto para ver cuál era la función a maximizar y cuáles iban a ser nuestros genes. Como el objetivo que se busca es optimizar el desempeño y éste desempeño depende de los ítems de vestimenta y la altura resultó fácil de ver que ésta función de desempeño era la función a maximizar y los genes iban a ser los ítems (los ids de arma, botas, casco, guantes y pecheras) y la altura.
Para modelar los individuos de la población se contemplaron distintas posibilidades. Inicialmente se pensó realizar una clase *Individuo* donde cada ítem y la altura serían atributos de ésta. Sin embargo, con esta opción, los métodos de cruza resultaban complejos de realizar debido a que cada gen se encontraba en una variable separada, por lo que se buscó una opción más eficiente. El resultado de esto fue la estructura que se terminó implementando que consta de una lista de diccionarios. Donde un individuo tiene un atributo height y una lista de id de ítems. Por ejemplo un individuo posible sería {'height': 1.564534, 'items': [10, 543, 12988, 98, 899888]}

El motor de algoritmos genéticos implementó:

*Operadores Genéticos:*

- Cruza:
  - Cruce de un punto
  - Cruce de dos puntos
  - Cruce uniforme
  - Cruce anular
- Mutación:
  - Mutación Gen
  - Mutación Multigen
  - Mutación Uniforme
  - Mutación No Uniforme

*Métodos de Selección:*

- Elite
- Ruleta
- Universal
- Boltzmann
- Torneos Deterministica
- Torneos Probabilistica
- Ranking

*Métodos de Reemplazo:*

- Método de Reemplazo 1
- Método de Reemplazo 2
- Método de Reemplazo 3

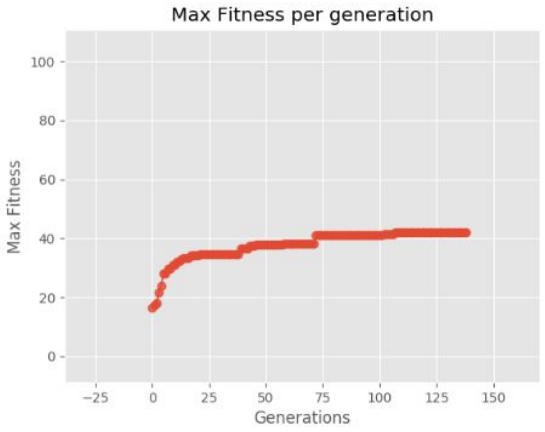
*Condiciones de Corte:*

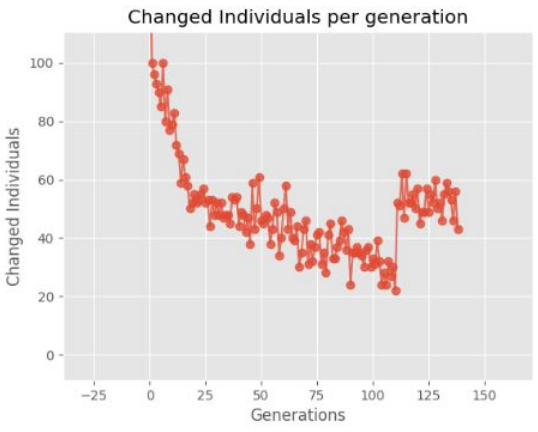
- Corte por contenido: El mejor fitness de la población no progresa con las generaciones
- Corte por Cantidad de Generaciones: Máxima cantidad de generaciones alcanzada
- Entorno a un óptimo: Se alcanza la solución óptima o se llega a un fitness inferior a una cota
- Corte por Estructura: Una parte relevante de la población no cambia de generación en generación

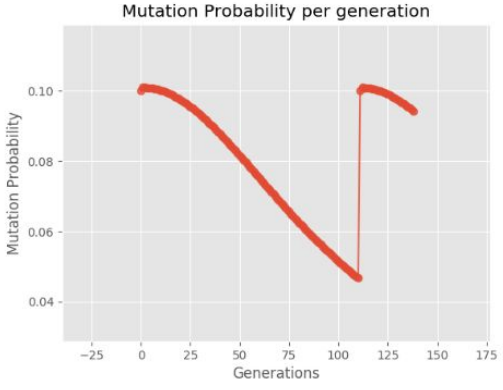
# Implementaciones adicionales

## Kicking:

Al realizar distintas pruebas con el motor de algoritmos genéticos se notó que al usar el método de mutación no uniforme la probabilidad de mutación llegaba a un valor tan bajo que el desempeño se estancaba en máximos locales. Al ver esto, se empezó a pensar en formas de poder salir de este estancamiento ya que el algoritmo podía tener información útil, o estar cerca de la solución y si se cortaba por estructura, esta información se iba a perder. Por este motivo surgió lo que llamamos corte Kicking. Este corte es una versión modificada del corte por estructura donde, en vez de cortar si hay pocos individuos nuevos, sube la probabilidad de mutación al valor inicial (“pega una patada”). Esto implica volver a la etapa de exploración pero si se utiliza junto con un método de selección como el Elite, los individuos de fitness deseado se mantendrán en la siguiente generación.



## Mutación Especializada:

La mutación convencional, como la tenemos implementada, elige un numero aleatorio en el dominio [1.3m ; 2m].

Se observó que por las característica de la altura de tener un dominio continuo, alcanzar la altura óptima (se demostrará más adelante que es 1.3m) a través de una mutación sería, probabilisticamente hablando, muy difícil.

Optamos entonces por generar una operación de mutación de altura un poco diferente, donde el individuo mutado obtendría una altura en el entorno de su altura original.

El algoritmo de mutación especializada determina el tamaño de la mutación en función de la cantidad de generaciones, comenzando en 0.1. A medida que el número de generaciones avanza el tamaño de la mutación disminuye en orden.

Se observó que utilizando este método, para el Defensor, con la configuración predeterminada, se alcanza el valor de altura 1.3 en aproximadamente 10 generaciones. Con la mutación estándar la altura tiende lentamente a 1.3 y en aproximadamente 100000 generaciones el valor de altura se estabiliza en un valor muy cercano a 1.3, aproximadamente 1.30001

## Método de Inicialización de Generación 0:

Para dar un mayor control sobre los genes presentes en la población en la primer generación se implementó la funcionalidad de Inicialización forzosa de la generación 0. Se dispone un archivo .tsv con los genes de los miembros de la generación en lugar de iniciarlos todos con valores aleatorios.

Se creyó en un principio que un buen uso de este método nos permitiría hacer un análisis más profundo de las diferencias entre los métodos de selección.

Sin embargo no se logró llegar a conclusiones enriquecedoras.

# Configuración de Parámetros para pruebas

La siguiente es la configuración predeterminada, usada para todas las pruebas salvo que se aclare lo contrario.

| Categoría | Átributo | Valor |
|---|---|---|
| Personaje | Tipo de Personaje | 'Defensor' |
| | Número de Personaje | 2 |
| | Altura Mínima | 1.3 |
| | Altura Máxima | 2 |
| Población | Población | 300 |
| | Población Reproductiva | 260 |
| Cruza | Método de Cruce | Cruce de dos Puntos |
| | Porcentaje de Cruce | 80% |
| Mutación | Tipo de Mutación | Multigen |
| | Uniformidad de Mutación | Uniforme |
| | Mutación Especializada de Altura | Verdadero |
| | Porcentaje de Mutación inicial | 0.1 |
| | Alpha de Enfriamiento | 0.0001 |
| Selección | Método de Selección 1 | Elite |
| | Método de Selección 2 | Boltzmann |
| | Método de Selección 3 | Elite |
| | Método de Selección 4 | Torneo Probabilístico |
| | A | 0.5 |
| | B | 0.5 |
| Boltzmann | Temperatura Inicial | 100 |
| | Temperatura Final | 23 |

| | | |
|---|---|---|
| | Programación de Enfriamiento | Multiplicativo Cuadrático |
| | Alpha de Enfriamiento | 0.97 |
| Torneo | Participantes | 3 |
| Reemplazo | Método de Reemplazo | Método 2 |
| Criterios de Corte | Generaciones Consecutivas | 100 |
| | Cantidad de Generaciones | 100,000 |
| | Entorno a un optimo: Optimo | 50 |
| | Entorno a un optimo: Delta | 1 |
| | Cambios despreciables: Factor despreciable | 0.02 |
| | Cambios despreciables: Delta de cambios | 2 |

# Selección de Parámetros

## Selección de Población:

### Tamaño de Población (N):

El tamaño de la población óptimo es el tamaño que es lo suficientemente grande como para poder albergar diversidad y a la vez lo suficientemente pequeño para que el algoritmo sea temporalmente eficiente.
Se realizaron pruebas con N = [30,90,150,210,300,350,500]. Los mejores resultados observados fueron con N=300.

### Tamaño de Población Reproductiva (K):

La elección de K se hizo en base a los resultados de las pruebas una vez fijado N=300. Se determinó que el K que permite alcanzar mejores desempeños en una cantidad razonable de generaciones es K=260.

## Selección de Cruza:

### Método de Cruza:

Se probaron todos los métodos de cruza con configuraciones similares, no se observaron grandes diferencias en el alcance de desempeños altos. Se decidió utilizar el Cruce uniforme.

## Selección de Mutación:

### Método de Mutación:

Se priorizó, durante la etapa de exploración, maximizar la diversidad genética. Con esto en mente, se eligió el método de mutación multigen por ser este el que provee la mayor cantidad de ella.

### Uniformidad de Mutación:

Se realizaron pruebas con probabilidad de mutación no uniforme. El espíritu detrás de ellas era lograr una clara distinción entre etapas de exploración (alta diversidad genética) y etapas de explotación (diversidad genética baja). Para lograr esto, a lo largo de las generaciones se disminuye la probabilidad de mutación. Esto genera que al principio se encuentre en una etapa de exploración y luego, en una de explotación. Sin embargo, se notó que utilizando la mutación uniforme y manteniendo el algoritmo en un estado constante de exploración se alcanzaban los mejores resultados de desempeño.

Mantener un nivel relativamente alto de mutación te garantiza diversidad genética. La cruza de individuos converge muy rápidamente al máximo que puede lograr. Lo único que trae nueva información a la población es la mutación.

### Porcentaje de Mutación:

Una vez tomada la decisión de mantener la uniformidad de mutación se probó diferentes valores de porcentaje de mutación a sabiendas que un bajo nivel llevaría a poca diversidad genética y un alto nivel sería similar a hacer random walk.
Se decidió entonces por utilizar el factor de mutación 0.1 por ser este el que mejor balance tiene entre ambos.

### Cooling alpha:

En las pruebas donde se utilizó mutación no uniforme se optó por un alpha de 0.0001, ya que se observó que la frecuencia con la que disminuye la probabilidad de mutación genera buenos resultados.

## Selección:

### Método de Selección:

Para la elección de los métodos de selección se priorizó la presencia del método 'Elite' que fue el que mejores resultados dió en un principio y el cual se sabe mejora el desempeño. Sin embargo, para evitar llegar a convergencias prematuras se combinó con otros métodos que proveían algo de aleatoriedad a la selección. Las pruebas demostraron que los mejores resultados eran dados por Boltzmann y el Torneo Probabilístico, ambos tienen la característica de, en cierta medida, reducir la posibilidad de caer en máximos locales.

### A y B:

Se eligió tanto para A como para B el valor 0.5 ya que este es el más coherente con nuestro razonamiento anterior y pruebas variando los valores de A y B dieron resultados con diferencias despreciables.

## Boltzmann:

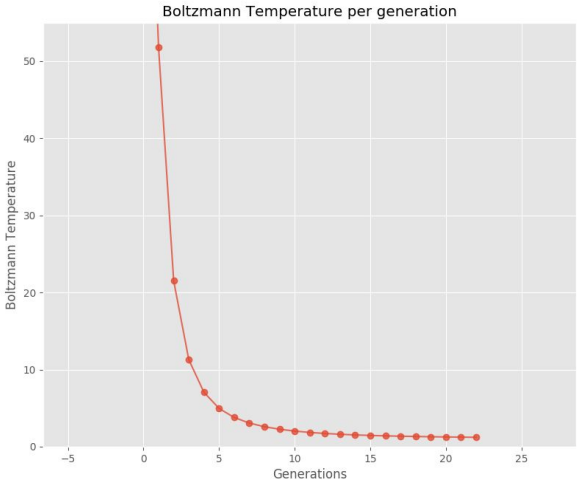
### Temperatura y Alpha

Para obtener la temperatura inicial y el alpha para boltzmann adecuado se utilizó un algoritmo de aproximación en el cual se determina la probabilidad de aceptación en dos momentos: inicialmente y en una generación arbitraria definida (en nuestro caso 200). Realizando un cálculo de esas dos probabilidades de aceptación definidas junto con el costo promedio de una transición se genera una estimación de la temperatura inicial y el alpha inicial. Los resultados obtenidos fueron: 93.92 para la temperatura y 0.9796 para el alpha.

Al ser esta una estimación no muy precisa, se realizaron también pruebas variando la temperatura para determinar cual se aplicaba mejor a nuestro caso en particular. Se encontró que una temperatura de 100 junto con un alpha de 0.97 generaban buenos resultados.

### Programaciones de Enfriamiento

Como vimos con Simulated Annealing, el método de enfriamiento que se elija tiene un efecto importante en los resultados que se obtienen. Por este motivo se decidió implementar varios para poder probarlos y encontrar el mejor.
Los que se implementaron fueron Quadratic Multiplicative, Logarithmical Multiplicative, Linear Multiplicative, Linear Additive y una adaptación que se encontró en internet[1].
Por la forma en la que decrece la temperatura con el método Quadratic Multiplicative, éste se eligió dentro de los cooling schedules.




# Reemplazo:

### Método de Reemplazo:

Entre los distintos métodos de reemplazo vistos en clase, el método 1 resultó el menos óptimo ya que no permite que ningún padre pase a la siguiente generación y de esta forma se puede estar perdiendo individuos valiosos. Por esto se decidió probar con el método 2 o 3 que poseen una brecha generacional distinta de 0.
Se decidió utilizar el método de reemplazo 2 ya que, además de tener menor costo computacional, resulta en una mayor cantidad de hijos cruzados y mutados que pasan a la generación siguiente. Esta característica implica que habrá mayor diversidad genética en la próxima generación ya que más individuos que quedan seleccionados fueron mutados, y, como se indicó previamente, la mutación trae información nueva a la población.

---

1 http://what-when-how.com/artificial-intelligence/a-comparison-of-cooling-schedules-for-simulated-annealing-artificial-intelligence/

## Criterios de Corte:

Para las pruebas se optó por utilizar el criterio de corte de contenido donde se corta cuando se observa que el algoritmo lleva una gran cantidad de generaciones sin mejorar su desempeño. Para asegurar esto y no cortar prematuramente, se eligió que la cantidad de generaciones consecutivas por la cual se permitiría continuar sin mejoras sería 1000000.

# Datos Obtenidos

El máximo desempeño alcanzado por nuestro algoritmo para el defensor 2 en todas las pruebas realizadas fue 50.44703720701878

La configuración del personaje óptimo fue:

- Altura:  1.3
- Vestimenta:
    - armas=167031
    - botas=801166
    - casco=158267
    - guantes=382596
    - pecheras= 849816

Esta configuración se alcanzó en la generación número 59730 y se mantuvo como la mejor hasta que se detuvo el proceso en la generación número 188850.

## Pruebas de Configuraciones

Se pusieron a prueba 9 configuraciones que mantenían en común los parámetros de la configuración predeterminada y añadían pequeñas modificaciones para evaluar los efectos de las mismas.

Todas las pruebas se dejaron correr en simultáneo por aproximadamente 16 horas.

Invariantes:

- Método de Cruza: ‘Uniforme’
- A y B: 0.5
- Boltzmann:
    - Temperatura inicial: 14
    - Temperatura final: 2
    - Alpha de enfriamiento: 0.97
    - Método de enfriamiento: 1
- Método de Reemplazo: 2

| | Parámetros | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Población | | Mutación | | | | Selección | Cruza |
| # | N | K | Uniformidad de Mutación | Probabilidad de Mutación inicial | Prob. Mut. alpha de enfriamiento | Geneticidad | Método de Selección | Prob. Cruza |
| 1 | 150 | 130 | Uniforme | 0.1 | N/A | Multi Gen | 1416 | 0.8 |
| 2 | 300 | 260 | Uniforme | 0.1 | N/A | Multi Gen | 1416 | 0.8 |
| 3 | 300 | 150 | Uniforme | 0.1 | N/A | Multi Gen | 1416 | 0.8 |
| 4 | 150 | 75 | Uniforme | 0.1 | N/A | Multi Gen | 1416 | 0.8 |
| 5 | 150 | 130 | No Uniforme | 0.1 | 0.0001 | Multi Gen | 1416 | 0.8 |
| 6 | 150 | 130 | No Uniforme | 0.3 | 0.00001 | Multi Gen | 1416 | 0.8 |
| 7 | 150 | 130 | Uniforme | 0.1 | N/A | Uni Gen | 1416 | 0.8 |
| 8 | 150 | 130 | Uni | 0.1 | N/A | Multi Gen | 1516 | 0.8 |
| 9 | 150 | 130 | Uniforme | 0.1 | N/A | Multi Gen | 1416 | 0.5 |

Configuración de las pruebas

| | Resultados | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Valores finales | | Generaciones para llegar a Umbrales de Desempeño | | | | | | | | | |
| # | Desempeño Max. | Generaciones Max. | 35 | 40 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 |
| 1 | 49.0065 | 35831 | 7 | 42 | 43 | 129 | 387 | 813 | 1213 | 2761 | 6429 | 13092 |
| 2 | 49.5119 | 18392 | 11 | 38 | 200 | 262 | 405 | 630 | 867 | 1464 | 4729 | 6678 |
| 3 | 49.2982 | 18590 | 12 | 43 | 102 | 102 | 125 | 193 | 852 | 3462 | 13023 | 16248 |
| 4 | 48.7133 | 36686 | 35 | 155 | 209 | 312 | 353 | 353 | 836 | 2166 | 15237 | N/A |
| 5 | 46.7922 | 36755 | 10 | 149 | 289 | 907 | 907 | 5293 | 466675 | N/A | N/A | N/A |
| 6 | 47.25 | 36481 | 11 | 78 | 139 | 147 | 196 | 11428 | 23839 | 29371 | N/A | N/A |
| 7 | 48.1935 | 35773 | 69 | 251 | 378 | 348 | 515 | 2296 | 3812 | 4706 | 30023 | N/A |
| 8 | 49.01 | 35742 | 12 | 14 | 77 | 145 | 225 | 286 | 697 | 1516 | 3500 | 10231 |
| 9 | 48.5152 | 36385 | 17 | 37 | 85 | 85 | 179 | 825 | 1330 | 2152 | 5865 | N/A |

Resultados de las pruebas

# Análisis de Resultados

## Mutación no Uniforme

Los resultados refuerzan la hipótesis que la mutación es la estrategia más efectiva para alcanzar desempeños altos.
Las configuraciones 5 y 6 difieren únicamente en el uso de mutaciones no uniformes, decrecientes. Se observa que el entrenamiento es similar durante las primeras generaciones pero al alcanzar aproximadamente las 1000 generaciones, cuando la probabilidad de mutación ya se redujo en casi dos órdenes de magnitud se converge a un desempeño relativamente bajo.
El hecho que la configuración con alpha de enfriamiento más bajo haya alcanzado un desempeño mayor refuerza el concepto de que la probabilidad de mutación es fundamental para evitar convergencias prematuras.

## Poblaciónes

Los resultados demuestran que la complejidad temporal aumenta linealmente con el valor de N, por lo cual aquellas con poblaciones altas alcanzaron valores de generación más pequeños.
A su vez se puede observar que las configuraciones con poblaciones más altas alcanzan mejores desempeños en un menor número de generaciones. Se podría decir que, ‘generacionalmente’, aprenden más rápido.
Invita a pensar que si bien el número de generaciones es menor, la cantidad de genes mutados y cruzas realizadas durante todo el entrenamiento es relativamente similar.
Los resultados muestran que valores de N y K más altos son preferibles.

## Altura óptima para el Defensor

Se calculó la derivada del desempeño del defensor en función de la altura (h).

$$\frac{\partial}{\partial h}\left(0.1\,F\left(-(3\,h-5)^4+(3\,h-5)^2+\frac{h}{2}+0.5\right)(A+P)+0.9\left((3\,h-5)^4-(3\,h-5)^2-\frac{h}{2}+2\right)(P+R)\,V\right)=(h^3-5\,h^2+8.27778\,h-4.53858)\,(291.6\,V\,(P+R)-32.4\,F\,(A+P))$$

Cálculo de la derivada del Desempeño en función de la altura

Calculando las raíces de la función g(h) = (h**3 - 5h**2 + 8.27778h -4.53858) se pueden conseguir las raíces de la derivada del desempeño que son independientes de los valores de los cinco atributos (V,P,R,F,A).

Roots:

$h \approx 1.44628$

$h \approx 1.63856$

$h \approx 1.91516$

Raíces de la función g(h)

Con estos valores de h sumados y los dos extremos del dominio (h=1.3 y h=2.0) se puede evaluar la función original en ellos para encontrar el máximo absoluto del desempeño en el dominio.

| h | Desempeño(h) |
|---|---|
| 1.30000 | $0.08959\,F\,(A+P)+1.44369\,V\,(P+R)$ |
| 1.44628 | $0.146919\,F\,(A+P)+0.927731\,V\,(P+R)$ |
| 1.63856 | $0.132634\,F\,(A+P)+1.05629\,V\,(P+R)$ |
| 1.91516 | $0.170447\,F\,(A+P)+0.715974\,V\,(P+R)$ |
| 2.00000 | $0.15\,F\,(A+P)+0.9\,V\,(P+R)$ |

Si bien esta información no es suficiente para asegurar que Desempeño(1.3) es el máximo absoluto (existen valores de los atributos para lograr desempeños mayores con alturas diferentes), sí se observa que el desempeño de 1.3 es el que logra maximizar el coeficiente que multiplica al sumando defensivo que es el más significativo en esta caso.

Esta información nos da seguridad que la tendencia del algoritmo a aproximarse al valor 1.3 de altura es acertada.

## Probabilidad de Cruza

Se observa en los resultados de la última configuración con una reducción en el valor de la probabilidad de cruza lleva a una convergencia prematura con respecto a las configuraciones con probabilidad de cruza más alta.

# Conclusión

Al implementar este motor de algoritmos genéticos se encontraron varias dificultades. Desde pensar la mejor estructura para la población, hasta encontrar las mejores configuraciones para el problema dado. Se fue descubriendo al investigar sobre los temas, que mucho del desarrollo y la obtención de estas configuraciones “ideales” se logra experimentalmente. Y de manera aún más relevante, el impacto de estas configuraciones en los resultados es, en algunos casos, drástico. Por estos motivos se buscó realizar muchas pruebas y utilizar algoritmos auxiliares (como en el caso de la temperatura inicial y alpha) para intentar obtener los parámetros que mejores resultados generen.

El principal problema encontrado en los resultados de las configuraciones probadas fue el de la convergencia prematura.

Se sabe por la teórica que algunas de las causas principales pueden ser

- Una presión de selección muy elevada
- Una probabilidad de mutación muy baja
- Un tamaño de población insuficiente

De todas los configuraciones empleadas para la reducción de la convergencia las que dieron los mejores resultados fueron aquellas que usaban una mutación uniforme relativamente alta (0.1).

También se notó que entre la cruza y la mutación, lo que mayor impacto tenía era la mutación. Intentando pensar el por qué de esto, se llegó a una posible conclusión que es que debido a la naturaleza de la función de desempeño y los valores de los ítems dados, un individuo con desempeño alto no implica que tenga ids de ítems que generen buenos resultados siempre. Es decir, puede que un individuo tenga un desempeño alto por tener un arma, bota y casco muy buenos pero guantes y pechera normal. Por lo que al cruzar este individuo y pasarle los genes de guantes y pechera a otro, el hijo creado no va a tener buen fitness. Sin embargo, al mutar el cambio en un individuo generalmente es de menor cantidad de genes que en la cruza. Adicionalmente, la mutación garantiza y mantiene un nivel de diversidad genética que la cruza no. Esto se debe a que luego de un cierto número de generaciones no hay nueva información genética (si se corre con probabilidad de mutación 0 y N=150, el algoritmo converge en 6 generaciones).

En este trabajo se buscó aprender de los errores del trabajo anterior y no satisfacerse con implementar solo lo pedido, sino intentar ir más allá y buscar soluciones creativas a los problemas encontrados. De esta manera surgieron las implementaciones adicionales como el método kicking, el análisis de temperatura inicial y alpha en boltzmann, el análisis de la altura y la posibilidad de determinar la población inicial para realizar pruebas. Estos nuevos desarrollos no fueron necesariamente soluciones ideales, sin embargo ayudaron a entender mejor el problema y ver en más detalle la evolución del algoritmo.