Metaheurísticas
Práctica 2

Algoritmos Genéticos

Damián Martínez Ávila

José Ángel Expósito Férnandez

Gabriel Cordero Contreras

1 de Abril de 2022

Índice

1. Codificación Binaria

1.1. Escalabilidad del Algoritmo

¿Cómo se comporta este algoritmo a medida que cambiamos el número de soluciones, generaciones, tamaño del torneo, probabilidad de cruce y probabilidad de mutación? Amplía el problema con más objetos para ver el comportamiento del algoritmo.

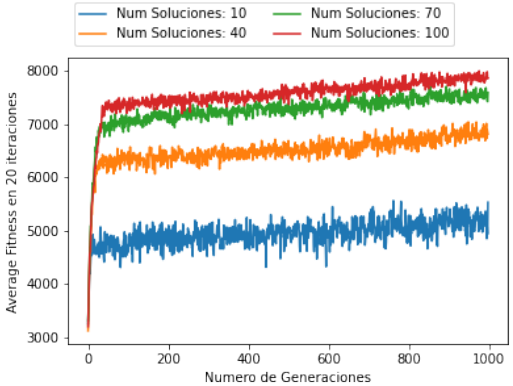
Vamos a proceder a calcular los parámetros óptimos para solucionar el problema de la mochila con 100 materiales con un peso entre 20 y 100 unidades. Para cada parámetro obtendremos el óptimo y lo mantendremos durante el cálculo del resto de parámetros, para obtener la combinación óptima.

1.1.1. Número de Soluciones

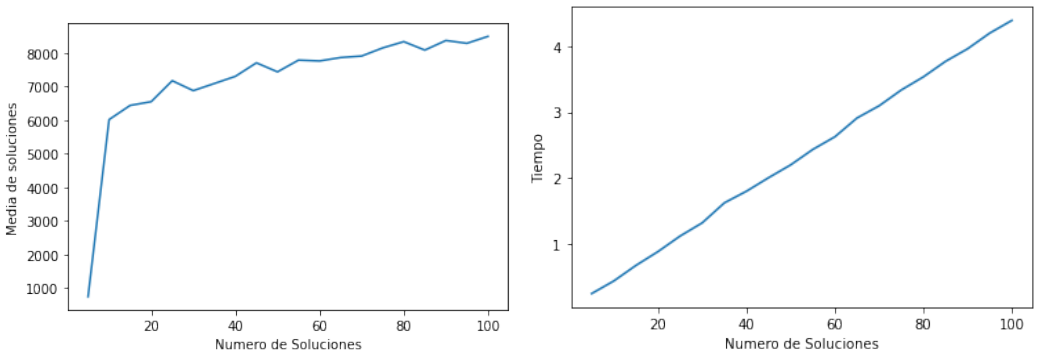
Parámetros

- Peso Máximo de la mochila: 1000
- Número de Generaciones: 1000
- Tamaño del torneo: 5
- Probabilidad de Cruce: 0.7
- Probabilidad de Mutación: 0.7

En la siguiente gráfica vemos como mejoran las poblaciones a medida que avanzan las generaciones para distintos valores del número de soluciones.



La gráfico de la izquierda muestra la media de las mejores soluciones en 20 iteraciones para valores del número de soluciones entre 10 y 100 con saltos de 5 en 5, y la de la derecha muestra como va incrementándose el tiempo.

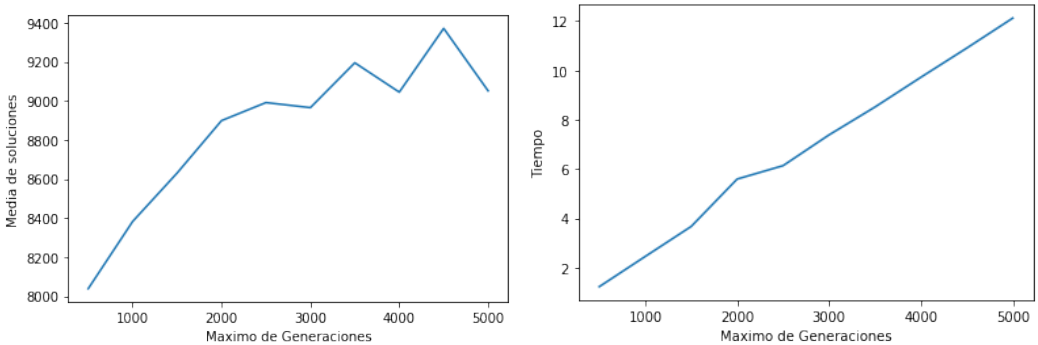


A la hora de elegir el parámetro de cara al número de soluciones vamos a elegir el valor 45, ya que da un resultado cercano al mejor en función del resultado (100) con un coste computacional mucho menor.

1.1.2. Número de Generaciones

Parámetros

- Peso Máximo de la mochila: 1000
- Tamaño de la población: 45
- Tamaño del torneo: 5
- Probabilidad de Cruce: 0.7
- Probabilidad de Mutación: 0.1

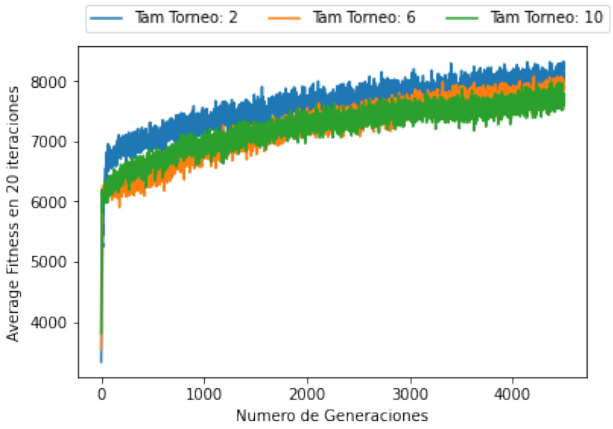


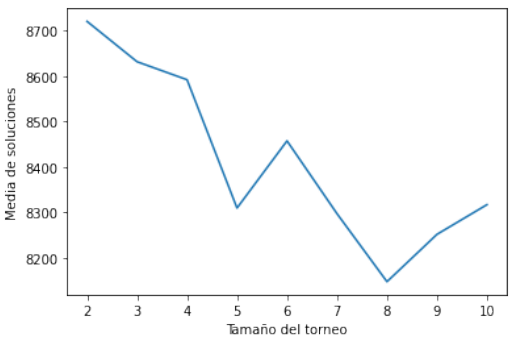
Vemos que a medida que aumentamos el número de generaciones obtenemos una mejor solución, si buscamos que converja en una solución cercana a la óptima, el número de generaciones debería ser muy elevado, en este caso escogeremos 4500 ya que nos da un mayor valor.

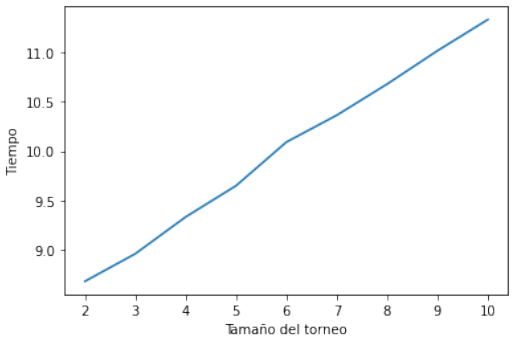
1.1.3. Tamaño del Torneo

Parámetros

- Peso Máximo de la mochila: 1000
- Tamaño de la población: 45
- Número de Generaciones: 4500
- Probabilidad de Cruce: 0.7
- Probabilidad de Mutación: 0.7







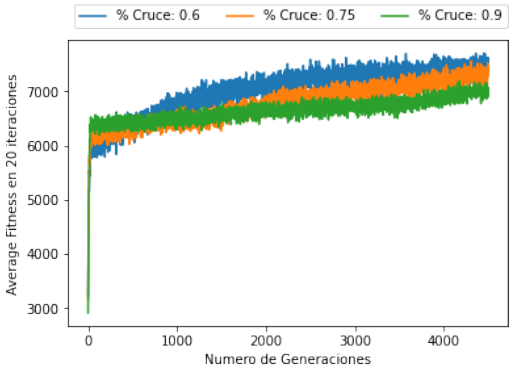
En cuanto al tamaño del torneo que usamos para seleccionar los padres, al aumentarlo podemos ver que la solución empeora debido a que intensifica la búsqueda en las mejores soluciones actuales y explora menor espacio de soluciones en la mayoría de los casos. Escogeremos por tanto un valor de 3.

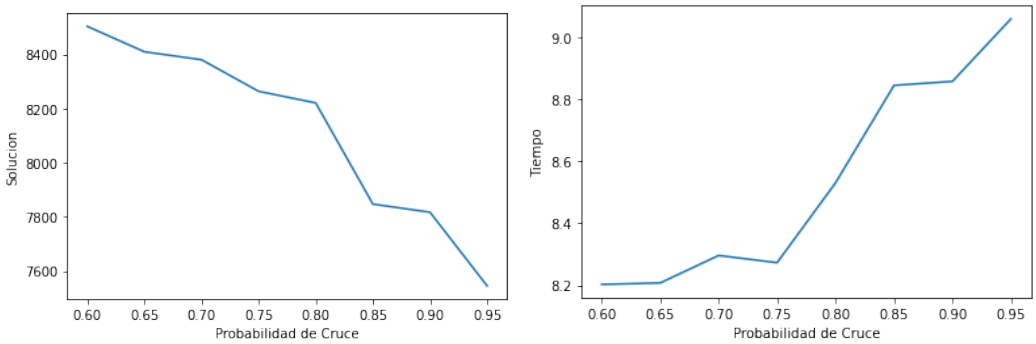
1.1.4. Probabilidad de Cruce

El cruce es un componente que tiende a intensificar las soluciones de las poblaciones. La implementación que hemos elegido es 1-cut, trocear dos soluciones en dos partes y combinarlas.

Parámetros

- Peso Máximo de la mochila: 1000
- Tamaño de la población: 45
- Número de Generaciones: 4500
- Tamaño del Torneo: 3
- Probabilidad de Mutación: 0.7





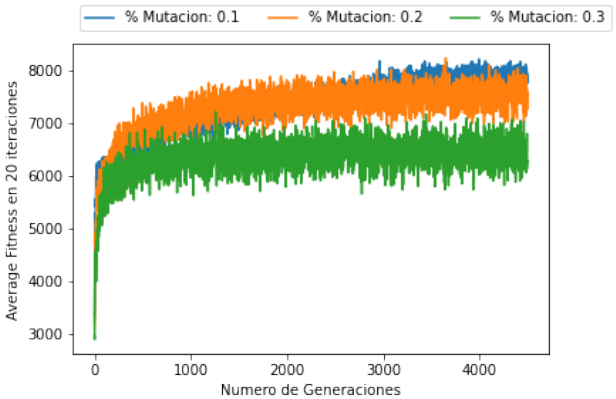
A la hora de elegir el mejor parámetro, vamos a elegir el valor 0.65 para que no intensifique ni diversifique la búsqueda en exceso, además obtenemos valores cercanos al óptimo y con un coste computacional bajo.

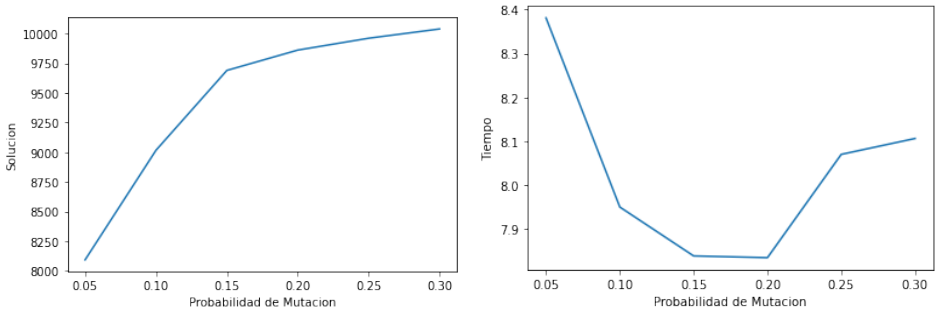
1.1.5. Probabilidad de Mutación

Es un componente que diversifica las soluciones de la población. La implementación se basa en sustituir un valor aleatorio de la solución.

Parámetros

- Peso Máximo de la mochila: 1000
- Tamaño de la población: 45
- Número de Generaciones: 4500
- Tamaño del Torneo: 3
- Probabilidad de Cruce: 0.65



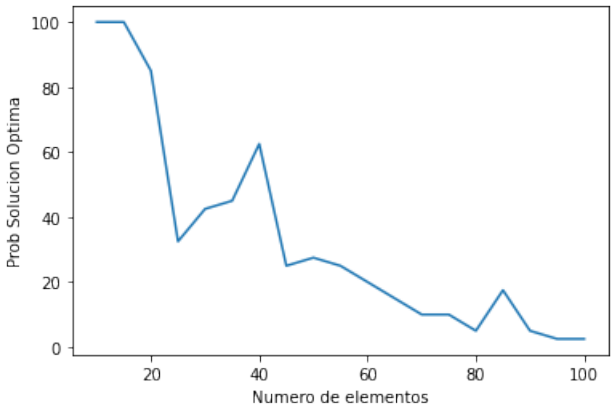


A la hora de elegir el mejor parámetro, vamos a elegir el valor 0.2 ya que aunque no sea el mejor valor respecto a los obtenidos, utilizar una mayor probabilidad de mutación generaría una mayor diversificación. En cuanto al tiempo, al ser una operación tan sencilla vemos que apenas hay diferencias entre los distinto valores.

1.2. Optimización de la Solución

¿Obtiene siempre la mejor solución? ¿Por qué? ¿De qué depende?

A medida que van aumentando el número de elementos, el problema se vuelve mas complejo, esto dificulta obtener la solución óptima. En la siguiente gráfica podemos observar este fenómeno ya que muestra la probabilidad de obtener la solución óptima(o una cercana con un margen del 1 %) respecto al número de elementos.

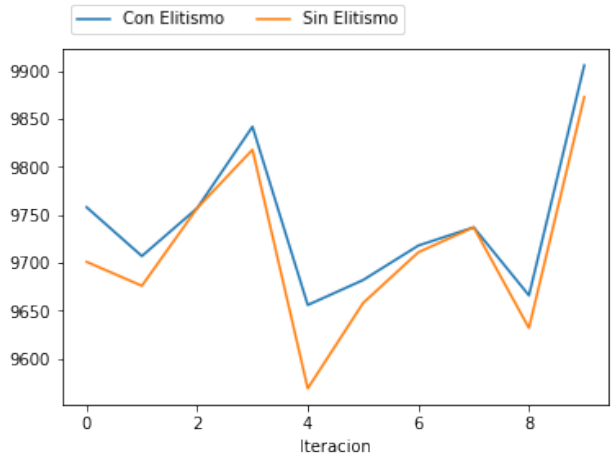


1.3. Elitismo

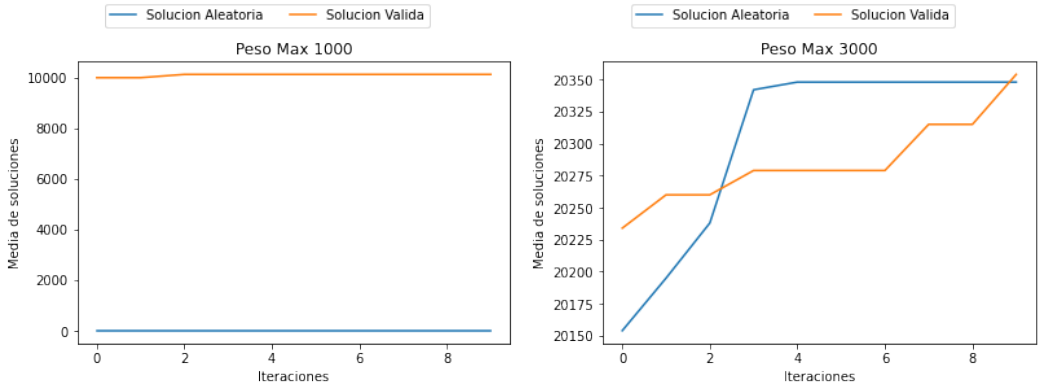
Modifica el código para incorporar elitismo. La mejor solución se guarda en la élite y nunca se pierde hasta que venga una nueva mejor. Esta solución es la que se devuelve al final. ¿Has conseguido mejorar? ¿Por qué?

Podemos ver que con elitismo en todos los casos mejora o mantiene el mismo resultado, esto es debido a que cuando no se usa elitismo hay cambios entre generaciones en los que se pierde la mejor solución al tener un valor del tamaño de torneos de selección de padres bajo.

En el caso que dicho valor fuera mas alto, se notaría menos el uso de elitismo, ya que con una mayor probabilidad pasaría la mejor solución a la siguiente generación.



1.4. Soluciones Iniciales Aleatorias



Podemos observar que en la primera gráfica, todas las soluciones son malas al iniciar de forma aleatoria, esto ocurre debido a que en la generación de nuestros materiales se le asignan pesos entre 20 y 100. Como las soluciones iniciales aleatorias tienen un 50 % de elegir un material, lo normal será tener soluciones con 50 materiales usados con un peso medio de 60. Por lo que el peso medio de una solución aleatoria sera de 3000, lo que supera ampliamente el peso máximo de 1000 y no encuentra ninguna solución válida, en la segunda gráfica y teniendo en cuenta esto, hemos puesto un peso máximo mayor y ahora si se encuentran soluciones válidas.

Para solucionar dicho problema, podríamos implementar que las evaluaciones de las soluciones puedan ser negativas, lo que castigará las peores soluciones. O se podría crear una mutación mayor que produzca soluciones más diversas si no mejora e intensifique al encontrar soluciones validas.

2. Codificación Entera

2.1. Escalabilidad del Algoritmo

¿Cómo se comporta este algoritmo a medida que cambiamos el número de soluciones, generaciones, tamaño del torneo, probabilidad de cruce y probabilidad de mutación? Amplía el problema con más objetos para ver el comportamiento del algoritmo.

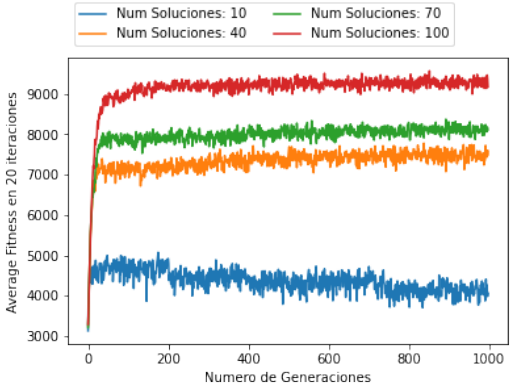
Vamos a proceder a calcular los parámetros óptimos para solucionar el problema de la mochila con 100 materiales con un peso entre 20 y 100 unidades. Para cada parámetro obtendremos el óptimo y lo mantendremos durante el cálculo del resto de parámetros, para obtener la combinación óptima.

2.1.1. Número de Soluciones

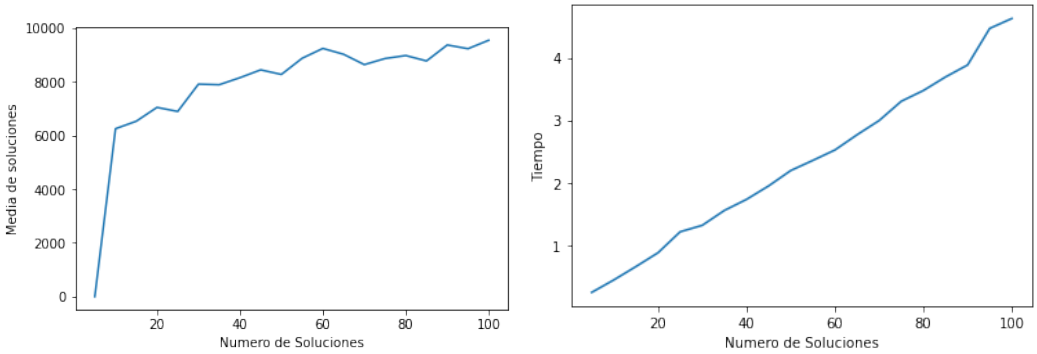
Parámetros

- Peso Máximo de la mochila: 1000
- Número de Generaciones: 1000
- Tamaño del torneo: 3
- Probabilidad de Cruce: 0.7
- Probabilidad de Mutación: 0.1

En la siguiente gráfica vemos como mejoran las poblaciones a medida que avanzan las generaciones para distintos valores del número de soluciones.



La gráfica de la izquierda muestra la media de las mejores soluciones en 20 iteraciones para cada valor del número de soluciones entre 10 y 100 con saltos de 5 en 5, y la de la derecha muestra como se incrementa el tiempo a medida que aumentamos el número de soluciones.



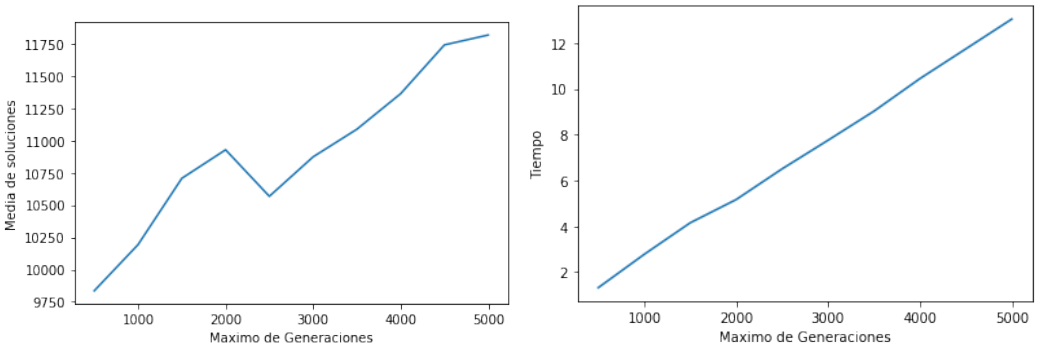
A la hora de elegir el parámetro de cara al número de soluciones vamos a elegir el valor 60, ya que da un resultado cercano al mejor en función del resultado (100) con un coste computacional mucho menor.

2.1.2. Número de Generaciones

Parámetros

- Peso Máximo de la mochila: 1000
- Tamaño de la población: 60
- Tamaño del torneo: 3
- Probabilidad de Cruce: 0.7
- Probabilidad de Mutación: 0.1

Vemos que a medida que aumentamos el número de generaciones obtenemos una mejor solución, hasta que se llega a un óptimo global.

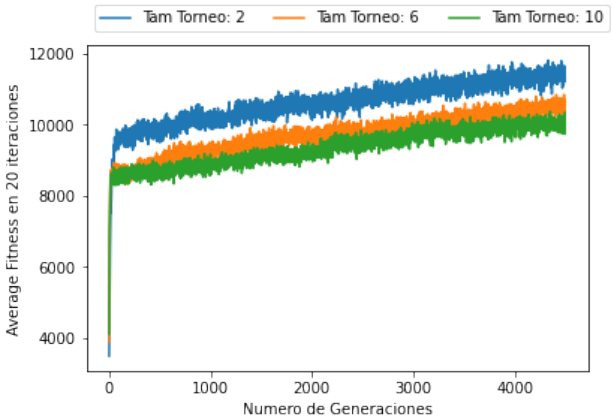


A la hora de elegir el mejor parámetro, vamos a elegir el valor 4500 ya que aunque no sea el mejor valor respecto a los posibles, es cercano al mejor valor (5000) pero con un coste computacional menor.

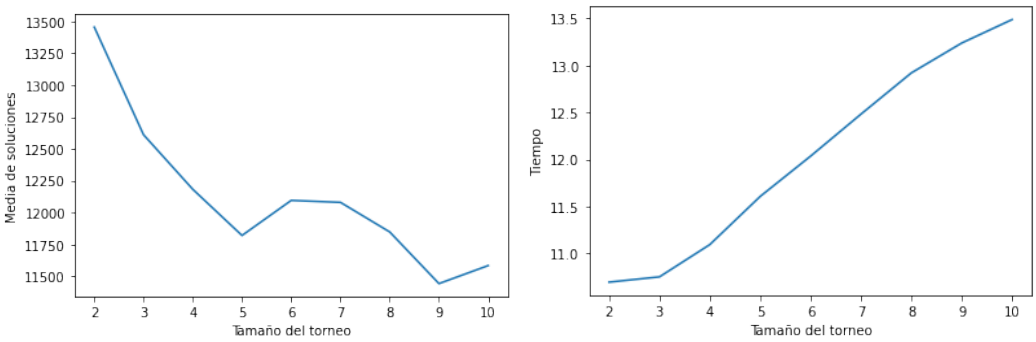
2.1.3. Tamaño del Torneo

Parámetros

- Peso Máximo de la mochila: 1000
- Tamaño de la población: 60
- Número de Generaciones: 4500
- Probabilidad de Cruce: 0.7
- Probabilidad de Mutación: 0.1



En cuanto al tamaño del torneo que usamos para seleccionar los padres, al aumentarlo podemos ver que la solución empeora debido a que intensifica la búsqueda en las mejores soluciones actuales y explora menor espacio de soluciones en la mayoría de los casos.



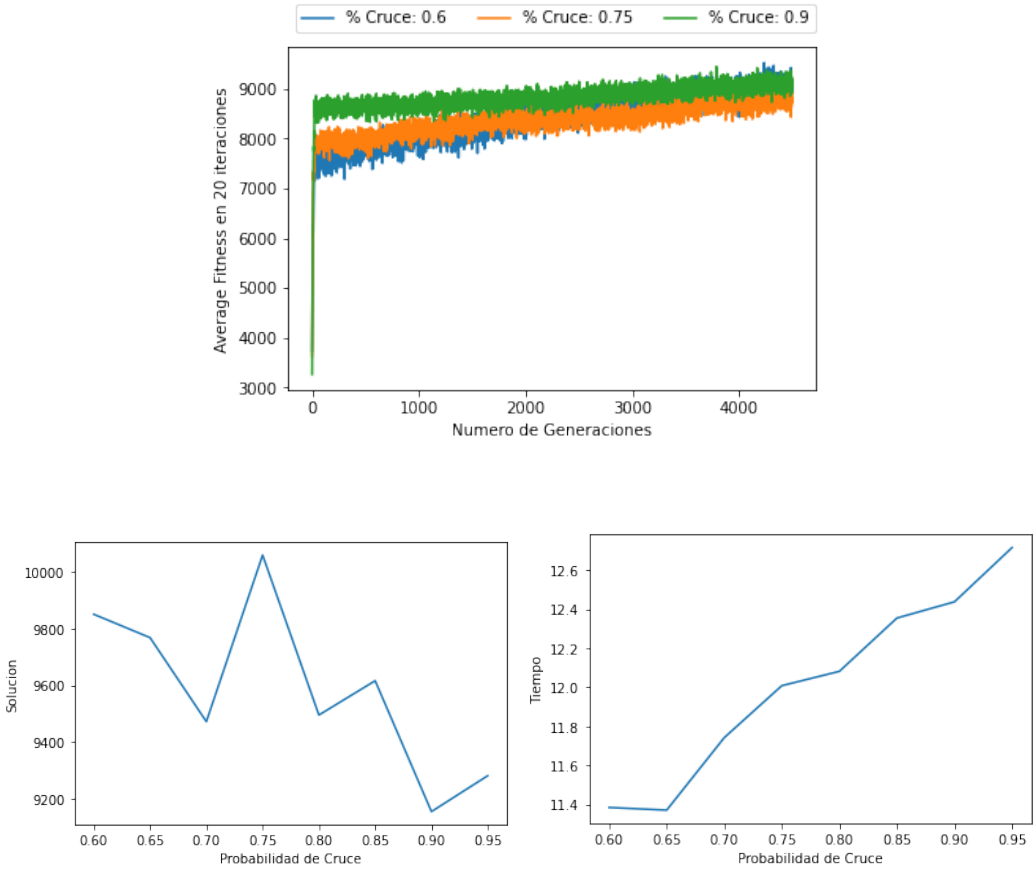
A la hora de elegir el mejor parámetro, vamos a elegir el valor 3 ya que aunque no sea el mejor valor respecto al mejor, esto hará que elija entre al menos tres opciones y con un coste computacional muy parecido a utilizar 2.

2.1.4. Probabilidad de Cruce

El cruce es un componente que tiende a intensificar las soluciones de las poblaciones. La implementación que hemos elegido es 1-cut, trocear dos soluciones en dos partes y combinarlas.

Parámetros

- Peso Máximo de la mochila: 1000
- Tamaño de la población: 60
- Número de Generaciones: 4500
- Tamaño del Torneo: 3
- Probabilidad de Mutación: 0.1



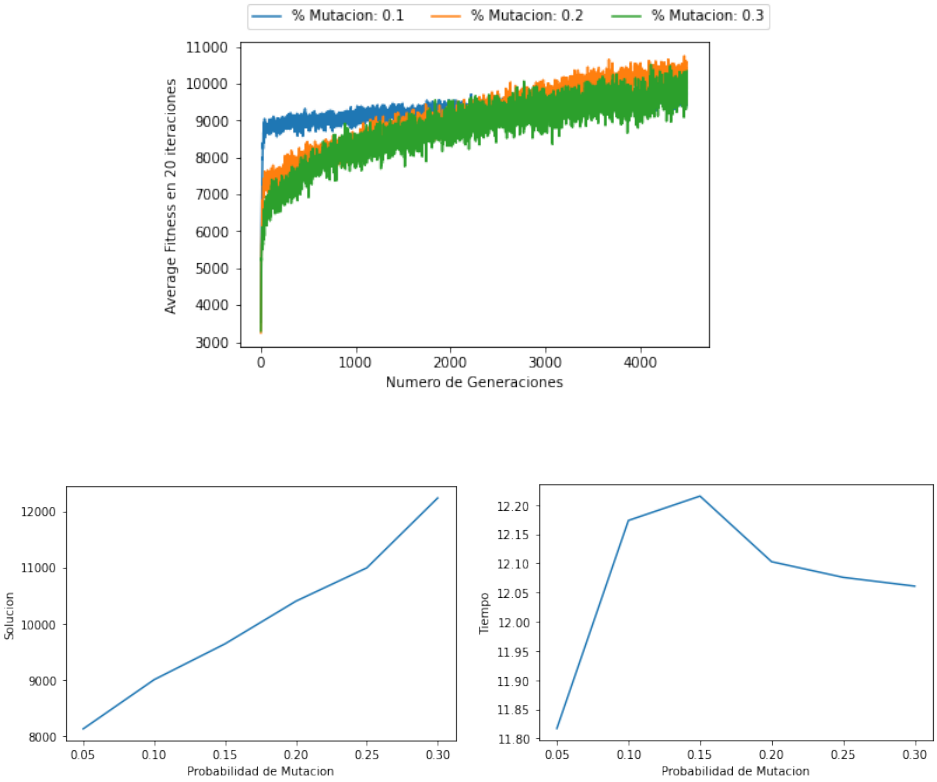
A la hora de elegir el mejor parámetro, vamos a elegir el valor 0.75 ya que es el mejor valor obtenido con mucha diferencia.

2.1.5. Probabilidad de Mutación

Parámetros

- Peso Máximo de la mochila: 1000
- Tamaño de la población: 60
- Número de Generaciones: 4500
- Tamaño del Torneo: 3
- Probabilidad de Cruce: 0.75

La mutación es un componente que diversifica las soluciones de la población. La implementación que hemos elegido trata de aumentar o restar una unidad a una posición aleatoria de la solución.



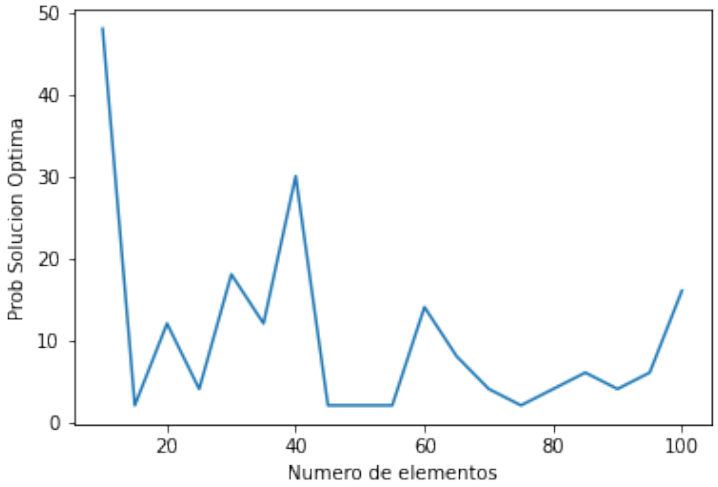
A la hora de elegir el mejor parámetro, vamos a elegir el valor 0.2 ya que aunque no sea el mejor valor respecto a los obtenidos, utilizar una mayor probabilidad de mutación generaría una mayor diversificación, que podría ralentizar la convergencia.

También hay que tener en cuenta que al no existir un limite de materiales y al realizarse tantas generaciones, puede darse el caso de que el algoritmo mute y elija los materiales con mayor rendimiento, esto se intensifica con un valor de mutación alto. En cuanto al tiempo, al ser una operación tan sencilla vemos que apenas hay diferencias entre los distinto valores.

2.2. Optimización de la Solución

¿Obtiene siempre la mejor solución? ¿Por qué? ¿De qué depende?

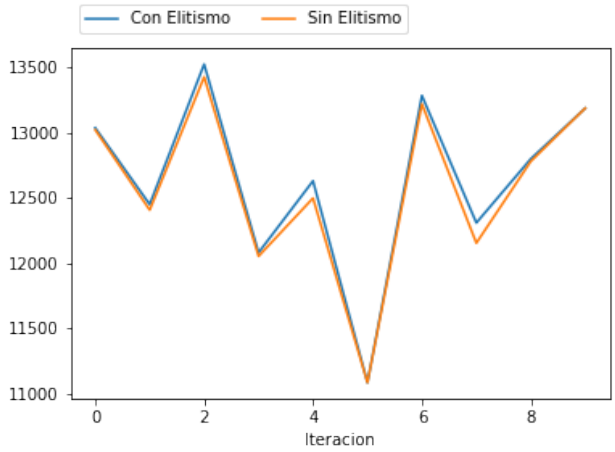
Con esta implementación el espacio de soluciones del problema aumenta, lo que genera que se obtengan resultados mas dispersos. Debido a esto obtener la solución óptima es muy complejo(aun aceptando como solución óptima soluciones con un margen del 5 %), lo cual se puede observar en el gráfico.



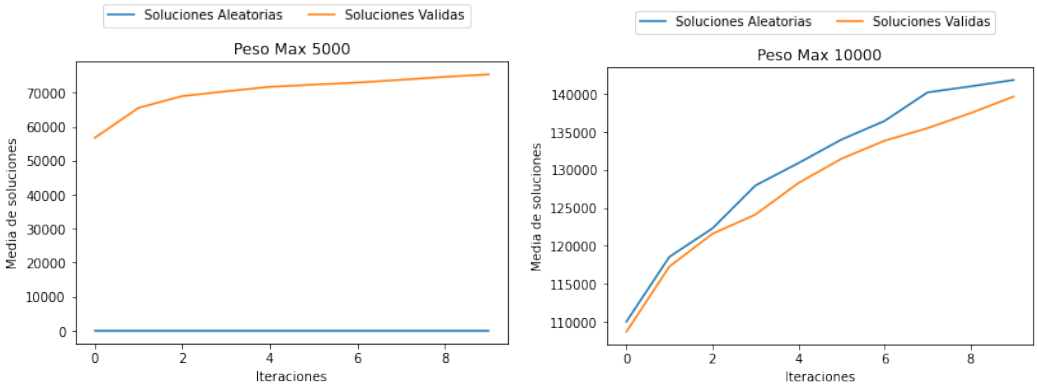
2.3. Elitismo

Modifica el código para incorporar elitismo. La mejor solución se guarda en la élite y nunca se pierde hasta que venga una nueva mejor. Esta solución es la que se devuelve al final. ¿Has conseguido mejorar? ¿Por qué?

Podemos ver que con elitismo en todos los casos mejora o mantiene el mismo resultado, esto es debido a que cuando no se usa elitismo hay cambios entre generaciones en los que se pierde la mejor solución al tener un valor del tamaño de torneos de selección de padres bajo. En el caso que dicho valor fuera mas alto, se notaría menos el uso de elitismo, ya que con una mayor probabilidad pasaría la mejor solución a la siguiente generación.



2.4. Soluciones Iniciales Aleatorias



Podemos observar que en la primera gráfica, todas las soluciones son malas al iniciar de forma aleatoria, esto ocurre debido a que en la generación de nuestros materiales se le asignan pesos entre 20 y 100, y que habrá entre 1 o 2 unidades de cada material en la mochila de 100 materiales. Por lo que el peso medio de una solución aleatoria sera de 9000, lo que supera ampliamente el peso máximo de 5000 y no encuentra ninguna solución válida, a diferencia de la segunda gráfica donde el peso máximo es de 10000 y entonces las soluciones generadas si están por debajo del límite.

Para solucionar dicho problema, podríamos implementar que las evaluaciones de las soluciones puedan ser negativas, lo que castigará las peores

soluciones. También se podría poner un límite en el número de veces que se pueda utilizar un material

Referencias

- Código Fuente Codificación Binaria: https://github.com/p82maavd/Metaheuristicas/blob/main/Practica2Metaheuristica_Codificacion_Binaria.ipynb
- Código Fuente Codificación Entera: https://github.com/p82maavd/Metaheuristicas/blob/main/Practica2Metaheuristica_Codificacion_Entera.ipynb