INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

EVOLUTIONARY COMPUTING

PRÁCTICA 4: INTRODUCCIÓN A ALGORITMOS GENÉTICOS

ALUMNO: ORTEGA VICTORIANO IVAN

PROF.: JORGE LUIS ROSAS TRIGUEROS

FECHA DE REALIZACIÓN DE LA PRÁCTICA: 03/04/2018

FECHA DE ENTREGA DEL REPORTE: 10/04/2018

MARCO TEÓRICO.

Entre los años 1950 y 1960, varios científicos en computación estudiaron independientemente sistemas evolutivos con la idea de que la evolución podría ser utilizada como una herramienta para la optimización en problemas de ingeniería. La idea en todos estos sistemas era evolucionar una población de soluciones candidatas de un problema dado, utilizando operadores inspirados por la variación genética natural y la selección natural. [1]

En los años 60's, Rechenberg introdujo "estrategias de evolución" (*Evolutionsstrategie* originalmente en alemán), un método que él utilizó para optimizar parámetros con valores reales para dispositivos tales como superficies aerodinámicas. Esta idea fue desarrollada más tarde por Schwefel. El campo de las estrategias de evolución ha permanecido un área activa de investigación, principalmente desarrollándose independientemente desde el campo de los algoritmos genéticos (aunque recientemente ambas comunidades han empezado a interactuar). [1]

Fogel, Owens y Walsh desarrollaron la "programación evolutiva", una técnica en la que soluciones candidatas a tareas dadas, fueron representadas como máquinas de estados finitos, las cuales evolucionaban mediante mutaciones aleatorias su diagrama de transiciones de estados y seleccionando su aptitud. Una formulación algo más amplia de la programación evolutiva también sigue siendo un área de investigación activa. Juntas, estrategias de evolución, programación evolutiva y los algoritmos genéticos forman los cimientos del campo del cómputo evolutivo. [1]

Los Algoritmos Genéticos son métodos adaptativos que pueden usarse para resolver problemas de búsqueda y optimización. Están basados en el proceso gen ético de los organismos vivos. A lo largo de las generaciones, las poblaciones evolucionan en la naturaleza de acorde con los principios de la selección natural y la supervivencia de los más fuertes, postulados por Darwin

en 1859. Por imitación de este proceso, los Algoritmos Genéticos son capaces de ir creando soluciones para problemas del mundo real. La evolución de dichas soluciones hacia valores óptimos del problema depende en buena medida de una adecuada codificación de las mismas. [2]

En la naturaleza los individuos de una población compiten entre sí en la búsqueda de recursos tales como comida, agua y refugio. Incluso los miembros de una misma especie compiten a menudo en la búsqueda de un compañero. Aquellos individuos que tienen más éxito en sobrevivir y en atraer compañeros tienen mayor probabilidad de generar un gran número de descendientes. Por el contrario, individuos poco dotados producirán un menor número de descendientes. Esto significa que los genes de los individuos mejor adaptados se propagarán en sucesivas generaciones hacia un número de individuos creciente. La combinación de buenas características provenientes de diferentes ancestros puede a veces producir descendientes “superindividuos”, cuya adaptación es mucho mayor que la de cualquiera de sus ancestros. De esta manera, las especies evolucionan logrando unas características cada vez mejor adaptadas al entorno en el que viven. [2]

Los Algoritmos Genéticos usan una analogía directa con el comportamiento natural. Trabajan con una población de individuos, cada uno de los cuales representa una solución factible a un problema dado. A cada individuo se le asigna un valor o puntuación, relacionado con la bondad de dicha solución. En la naturaleza esto equivaldría al grado de efectividad de un organismo para competir por unos determinados recursos. Cuanto mayor sea la adaptación de un individuo al problema, mayor será la probabilidad de que el mismo sea seleccionado para reproducirse, cruzando su material genético con otro individuo seleccionado de igual forma. Este cruce producirá nuevos individuos (descendientes de los anteriores), los cuales comparten algunas de las características de sus padres. Cuanto menor sea la adaptación de un individuo, menor será la probabilidad de que dicho individuo sea seleccionado para la reproducción, y por tanto de que su material genético se propague en sucesivas generaciones. [2]

De esta manera se produce una nueva población de posibles soluciones, la cual reemplaza a la anterior y verifica la interesante propiedad de que contiene una mayor proporción de buenas características en comparación con la población anterior. Así a lo largo de las generaciones las buenas características se propagan a través de la población. Favoreciendo el cruce de los individuos mejor adaptados, van siendo exploradas las áreas más prometedoras del espacio de búsqueda. Si el Algoritmo Genético ha sido bien diseñado, la población convergerá hacia una solución óptima del problema. [2]

El poder de los Algoritmos Genéticos proviene del hecho de que se trata de una técnica robusta, y pueden tratar con éxito una gran variedad de problemas provenientes de diferentes áreas, incluyendo aquellos en los que otros métodos encuentran dificultades. Si bien no se garantiza que el Algoritmo Genético

encuentre la solución óptima del problema, existe evidencia empírica de que se encuentran soluciones de un nivel aceptable, en un tiempo competitivo con el resto de algoritmos de optimización combinatoria. En el caso de que existan técnicas especializadas para resolver un determinado problema, lo más probable es que superen al Algoritmo Genético, tanto en rapidez como en eficacia. El gran campo de aplicación de los Algoritmos Genéticos se relaciona con aquellos problemas para los cuales no existen técnicas especializadas. Incluso en el caso en que dichas técnicas existan, y funcionen bien, pueden efectuarse mejoras de las mismas hibridándolas con los Algoritmos Genéticos. [2]

MATERIAL Y EQUIPO.

- Equipo de cómputo (PC o laptop).
- Python2 o Python3

DESARROLLO DE LA PRÁCTICA.

Una vez descargado el script de Python que se proporcionó en la página de wikispaces, procedimos a interactuar con él, ejecutándolo y observando su comportamiento. De lo que se pudo apreciar, se trataba de un algoritmo genético que trataba de encontrar un mínimo de una función, en la figura 1 podemos ver que se trata de una ventana en la que se tiene una gráfica de una función definida en el script, además de un grupo de líneas verticales que intersecan a la función, las cuáles podemos decir que es la población inicial utilizada en el algoritmo genético.

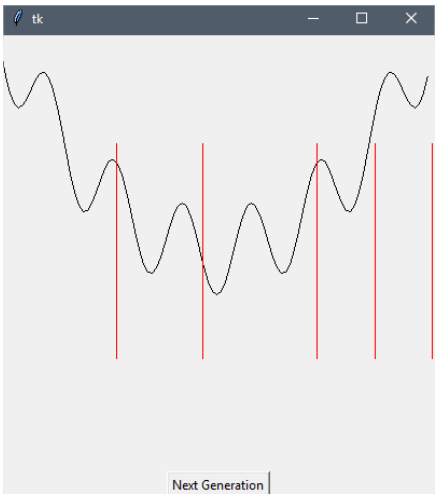

Figura 1. Captura del script en ejecución.

Al presionar el botón “Next Generation”, se generaba una nueva población siguiendo las condiciones del algoritmo genético canónico. Lo que se pudo experimentar en clase, fue que por más que se hicieran nuevas generaciones, el algoritmo se estancaba en una solución que no era la óptima, como se muestra en la figura 2.

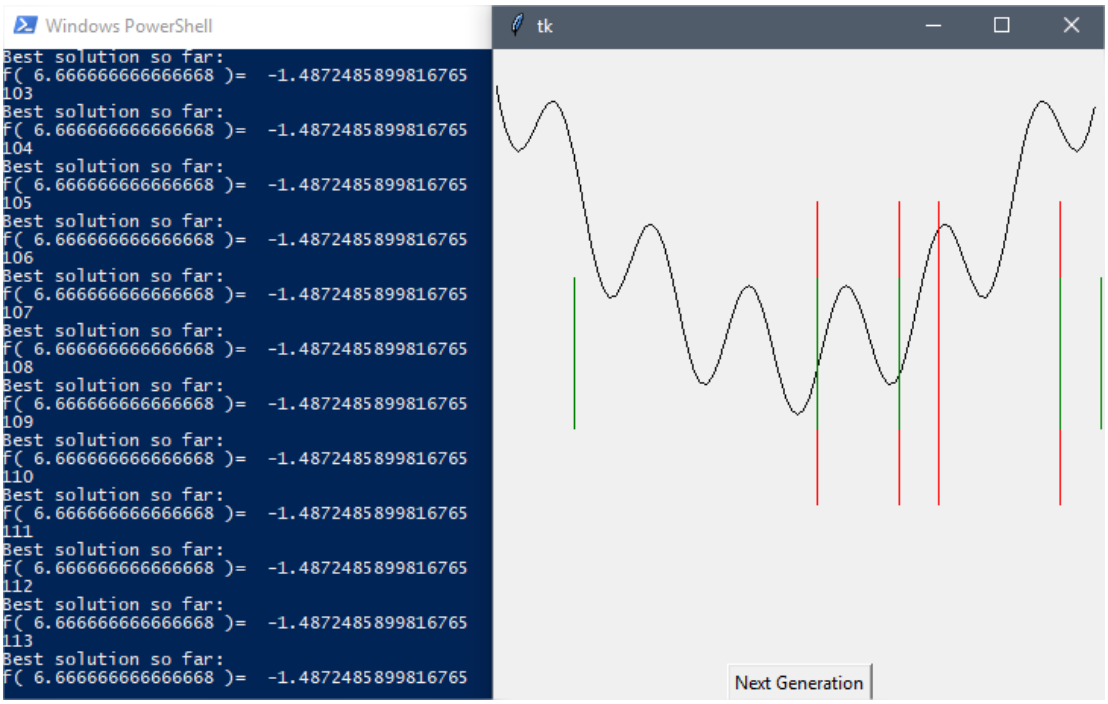

Figura 2. Después de más de 100 generaciones no se observa cambio alguno en la solución encontrada por el algoritmo.

Debido a lo anterior, se nos indicó que cambiáramos el valor de la longitud del cromosoma a un valor mayor, su valor inicial fue de 4, al cambiarlo a 5 hubo una mejora que se aproximaba un poco más a la solución óptima pero que aún estaba un poco lejos del valor ideal como se muestra en la figura 3.

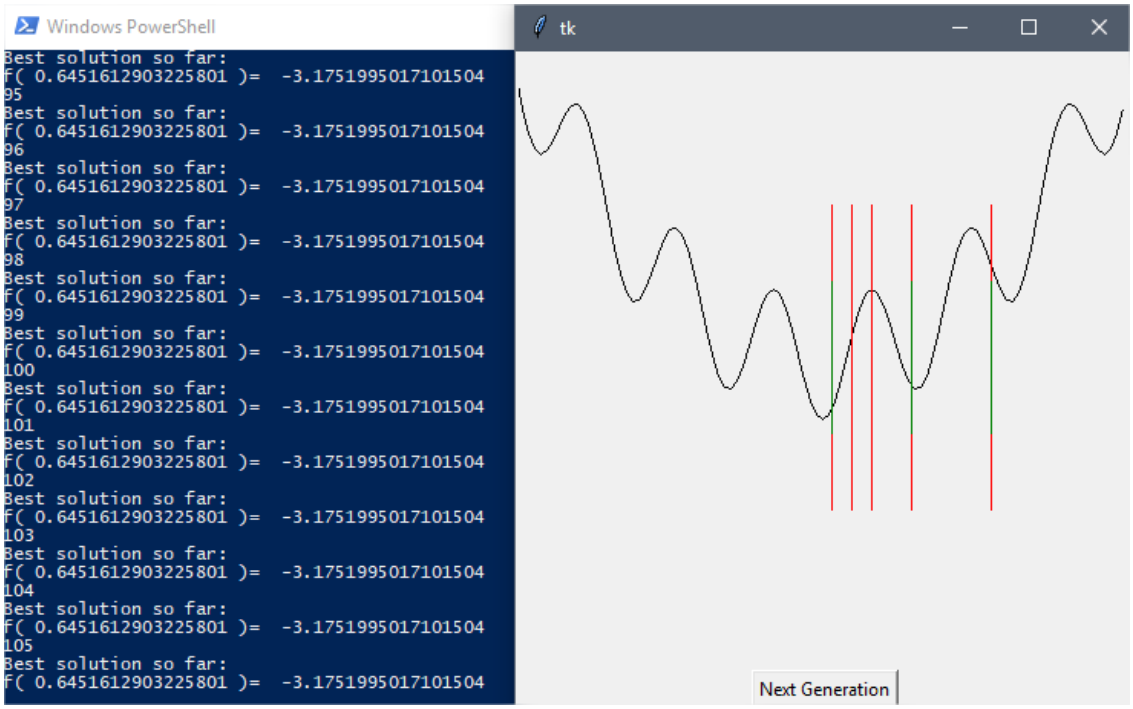

Figura 3. Resultado modificando la longitud del cromosoma a 5.

Conforme fuimos aumentando el valor de la longitud del cromosoma observamos que la solución encontrada por el algoritmo se aproximaba aún más a la solución real, en la figura 4 se puede ver que el resultado obtenido es cada vez más próximo a la solución (0,-4). En la figura 5 podemos ver que tardó 20 iteraciones en converger a un valor cercano a la solución óptima.

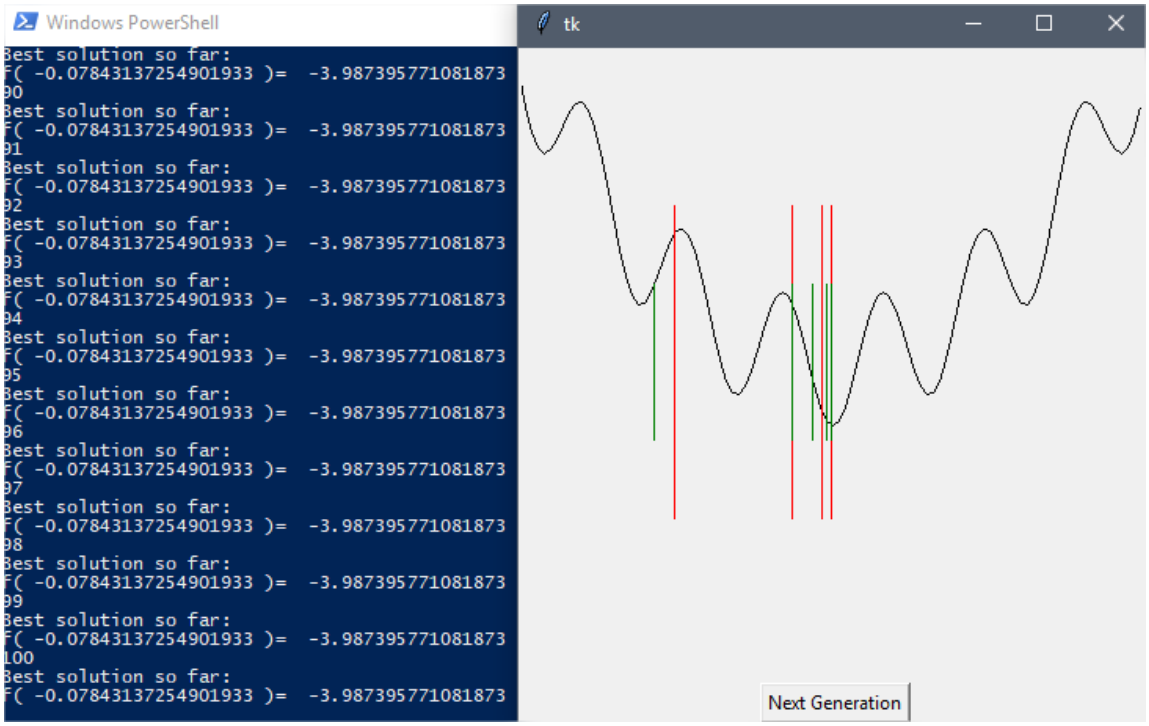

Figura 4. Resultado obtenido para una longitud de cromosoma igual a 8.

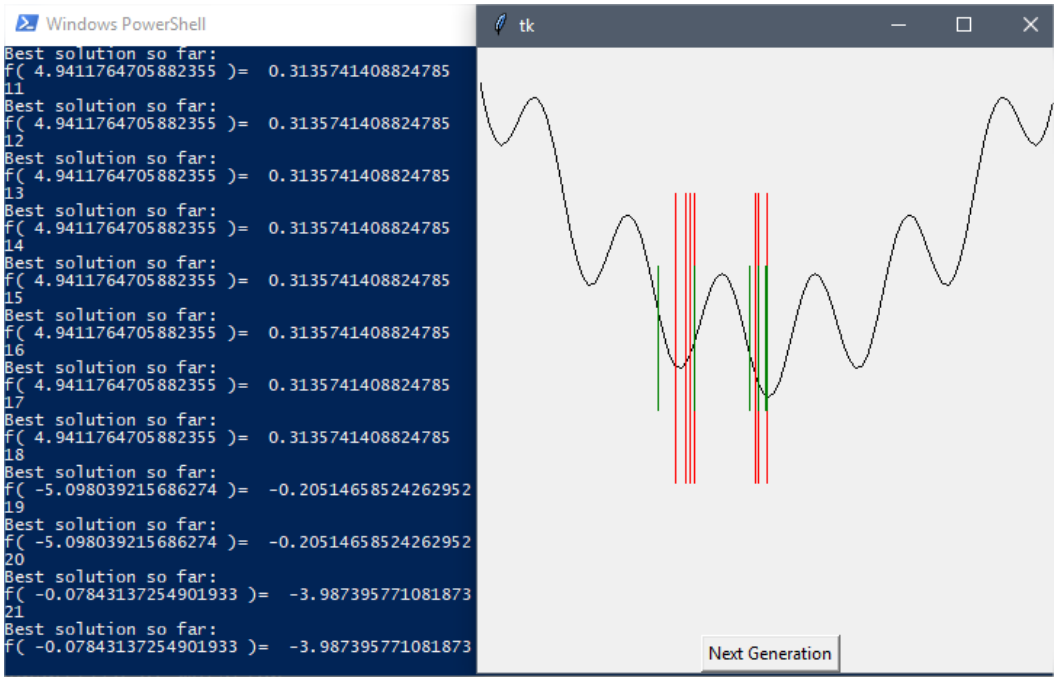

Figura 5. Convergencia a la solución óptima.

Posteriormente se nos solicitó modificar el valor del número de cromosomas y observar el comportamiento. Como se muestra en la figura 6, en ocasiones el algoritmo llegaba a converger a la solución en tan solo una generación. Mientras que en la figura 7 podemos ver que tardó unas cuantas generaciones más para converger.

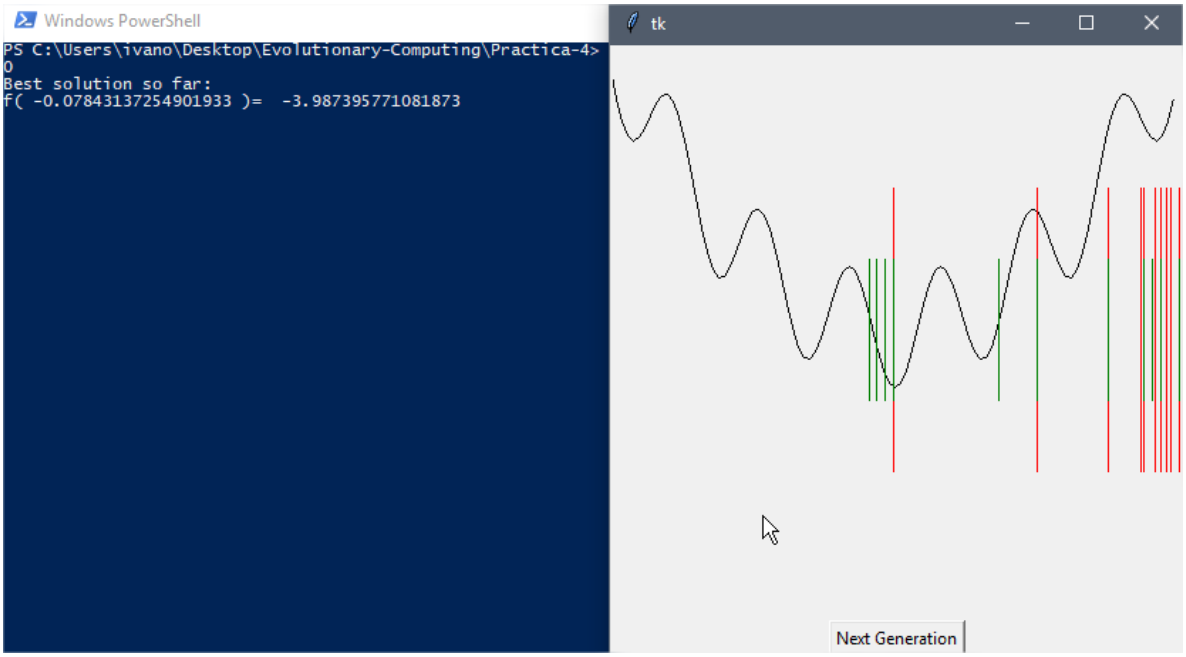

Figura 6. Convergencia a la solución óptima en solo una generación.

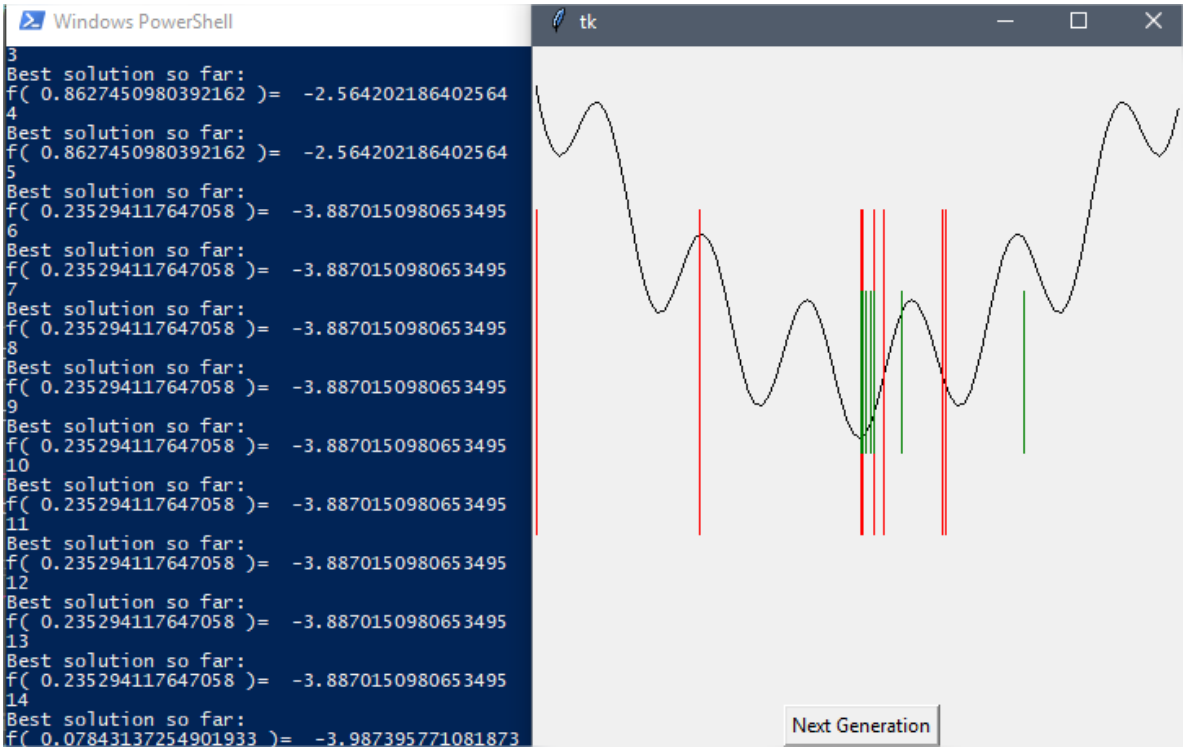

Figura 7. Convergencia a la solución óptima en 14 generaciones.

Al modificar el parámetro de número de cromosomas era más probable que el algoritmo convergiera a la solución en menos generaciones, de hecho, mientras más individuos se propongan en un algoritmo genético, mayor es la probabilidad de hallar la solución óptima, sin embargo, esto tendería a convertirse en una solución por fuerza bruta o búsqueda exhaustiva, pues los individuos explorarían casi todas las posibles soluciones para poder encontrar la óptima, lo cual, no es el objetivo de esta heurística.

Por otro lado, se nos pidió modificar el valor de la probabilidad de mutación. Dicho valor permite a individuos de la nueva generación obtener nuevas características, lo que en nuestro algoritmo implica explorar nuevas regiones que posiblemente no hayan sido exploradas en caso de que sea un valor muy grande (cercano a 1). En cambio, si la probabilidad es muy pequeña (cercana a 0), se les restringe a los nuevos individuos explorar nuevas regiones, por lo que es probable que la convergencia a la solución sea más tardada, o en el peor de los casos, que nunca se converja.

Empezamos probando con un valor de probabilidad de mutación de 0.1, una longitud de cromosoma de 8 y un número de cromosomas igual a 4. En la figura 8 podemos ver que, a pesar de haber generado 300 generaciones, la convergencia a la solución óptima no se ha dado.

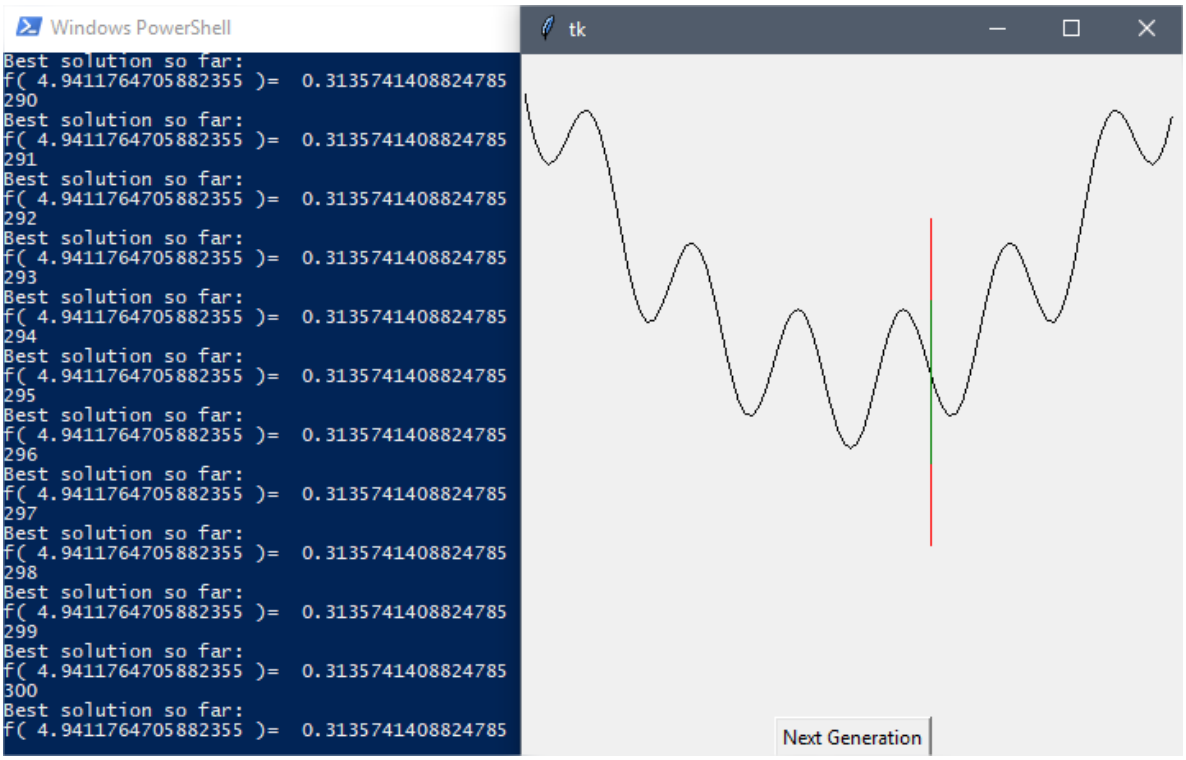

Figura 8. Resultados obtenidos después de 300 generaciones con una probabilidad de mutación baja.

En cambio, para los mismos valores, pero una probabilidad de mutación más alta de 0.9, podemos ver que es posible converger a la solución óptima en menos generaciones como se puede ver en la figura 9.

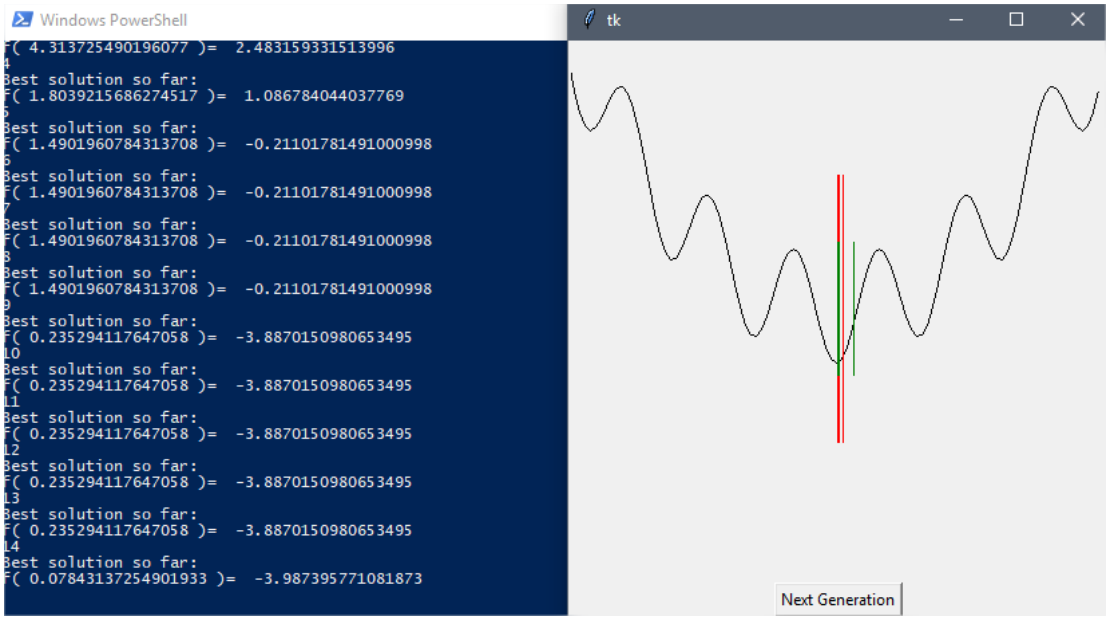

Figura 9. Resultados obtenidos con una probabilidad de mutación más alta.

Finalmente, se nos dejó experimentar con el programa, pero ahora con las funciones de Rastrigin y de Ackley.

Para dichos problemas, se utilizaron las funciones implementadas para la práctica 1. En la figura 10 podemos observar los resultados obtenidos para la función de Rastrigin, mientras que en la figura 11 podemos ver los resultados para la función de Ackley.

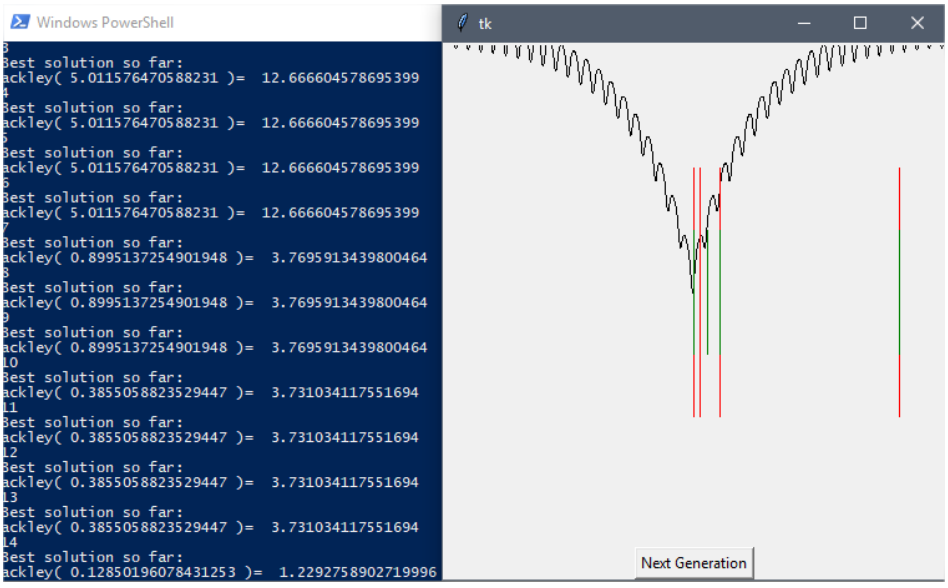

Figura 10. Resultados del algoritmo genético para la función de Ackley con una longitud de cromosoma 8, 10 cromosomas y probabilidad de mutación de 0.5.

Figura 11. Resultados del algoritmo genético para la función de Rastrigin con una longitud de cromosoma 8, 10 cromosomas y probabilidad de mutación de 0.5.

Como podemos ver, en el caso de la función de Rastrigin se obtuvo un valor más cercano al mínimo global que es (0,0) para su versión 1-D convergiendo en 20 generaciones a la tupla (0.02007,0.07987) aproximadamente (como se mostró en la figura 11). En cambio, la función de Ackley (figura 10), el valor más cercano al que se pudo llegar fue a (0.12,1.22) aproximadamente, que está un poco lejos de (0,0) que es la solución óptima.

CONCLUSIONES.

En un principio me fue difícil comprender el funcionamiento del algoritmo genético, ya que no había tratado con ellos previamente. Sin embargo, después de la clase de teoría pude entender mejor como es que trabajan. De lo que pude encontrar, fue que este tipo de algoritmos son utilizados en problemas donde no se tiene forma de hallar la solución de un problema en tiempo polinomial (generalmente problemas NP), así, este tipo de algoritmos son utilizados como una heurística para hallar una solución cercana a la óptima en un tiempo de cómputo razonable. En alguna ocasión escuché a un compañero hablar sobre cómo los algoritmos genéticos pueden ser utilizados para encontrar los valores cercanos a los óptimos de las matrices de pesos y los vectores bias en redes neuronales, lo cual me resultó muy interesante, ya que el algoritmo implementado para dicha tarea es muy tardado, aunque su precisión es muy buena su complejidad aumenta de acuerdo con el tamaño de la red.

En lo personal, este tipo de algoritmos me resultan muy interesantes. El hecho de que se basen en elementos o comportamientos de la naturaleza hace que su

potencial sea mucho mayor, lo cual permite que puedan aplicarse en distintas áreas debido a su adaptabilidad.

En cuanto a los códigos, tuve que realizar unas ligeras modificaciones para poder ejecutarlos en Python 3, además, pude darme cuenta de que al algoritmo genético le costó mucho más converger a la solución óptima de la función de Ackley, ya que para valores mayores a 0.1 o menores a -0.1, la función crece muy rápido a partir de ese punto llegando a puntos en los que hay crecimientos y decrecimientos y así sucesivamente.

Referencias

[1] M. Melanie, An Introduction to Genetic Algorithms, London, England: MIT Press, 1999.

[2] [En línea]. Available: http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/temageneticos.pdf.