Universidad del Valle de Guatemala
Modelos y Simulación - sección 10
Oscar Juárez - 17315; José Cifuentes - 17509
Fecha: 25/11/2020

Algoritmos genéticos

Historia

Los inicios de los algoritmos genéticos se remontan a finales de los años 50 y principios de los 60, en donde el biólogo Alexander Fraser publicó una serie de trabajos sobre la evolución de los sistemas biológicos en una computadora digital. Estos trabajos incluyen el uso de una reproducción binaria, así como también un operador de reproducción probabilística en donde a partir de una población de padres se generaba una nueva población de hijos tras combinarse y el mecanismo de selección. Estos trabajos serían la inspiración para lo que más tarde se llamaría Algoritmos genéticos (Villegas, 2005).

Años más tarde, en los años 60, Hans-Joachin Bremermann publicó una serie de artículos en donde adopta una población de solución a problemas de optimización, sometidos a reproducción, mutación y selección. La investigación de Bremermann incluyó los elementos de los algoritmos genéticos modernos. Él fue el primero en ver la evolución como un proceso de optimización (Berzal, 2014).

Finalmente, los algoritmos genéticos se hicieron populares gracias al trabajo y el libro Adaptation in Natural and Artificial System de John Henry Holland a principios de los años 70. Estas investigaciones permanecieron en gran parte teóricas hasta mediados de los años 80 cuando la primera conferencia internacional en algoritmos genéticos fue llevada a cabo en Pensilvania (Berzal, 2014).

Codificación

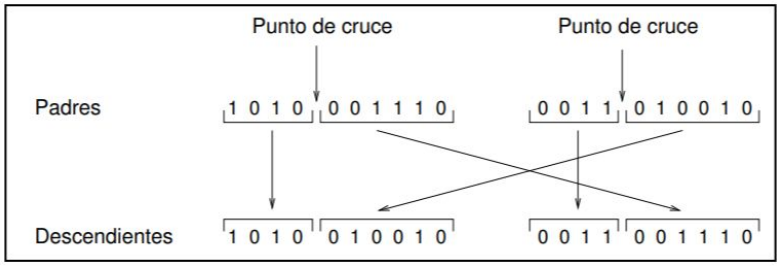
Para la resolución de problemas con algoritmos genéticos se utiliza una serie de conceptos claves que a continuación se explicarán. Primero, las posibles soluciones del problema pueden representarse como un conjunto de parámetros, estos se denominan genes. Si estos genes son agrupados, entonces se les llaman cromosomas. Buena parte de la teoría en la que se fundamentan dichos algoritmos utilizan solo ceros y unos para representar a los individuos, sin embargo, no necesariamente se tiene que utilizar este alfabeto (UPV, 2003).

La **función de adaptación** es propia de cada problema y esta tiene la tarea de a partir de un cromosoma retornar un número real el cual debería de reflejar el nivel de adaptación al problema. La escala en la que se mide el nivel de adaptación al problema también queda a discreción del programador (UPV, 2003).

En la **fase reproductiva** se seleccionan los individuos para producir descendientes que una vez mutados serán la siguiente generación. El proceso de selección de los padres está basado en la ruleta sesgada, es decir que es un proceso aleatorio que favorece a los mejores adaptados, es decir que los individuos mejores adaptados tienen mayor probabilidad de ser seleccionados (UPV, 2003).

El **operador de cruce** básico se realiza con una probabilidad entre 0.5 y 1. En el caso en el que no se aplique este procedimiento, los descendientes serán copias exactas de los padres. El operador básico consiste en seleccionar un punto aleatorio en donde se va a cortar la ristra. Posteriormente, se intercambian las subristras finales y así se producen los descendientes, podemos observar el procedimiento en la Figura 1 (UPV, 2003).

Figura 1: Operador de cruce basado en un punto.



El **operador de mutación** básico se realiza normalmente con probabilidades muy pequeñas. Este consiste en alteraciones aleatorias de cada gen que conforman el cromosoma. Esto asegura que ningún punto del espacio de soluciones tenga probabilidad cero de ser examinado y es de suma importancia para asegurar la convergencia de los algoritmos genéticos. En la figura 2 podemos observar el operador de mutación básico (UPV, 2003).

Figura 2: Operador de mutación básico.

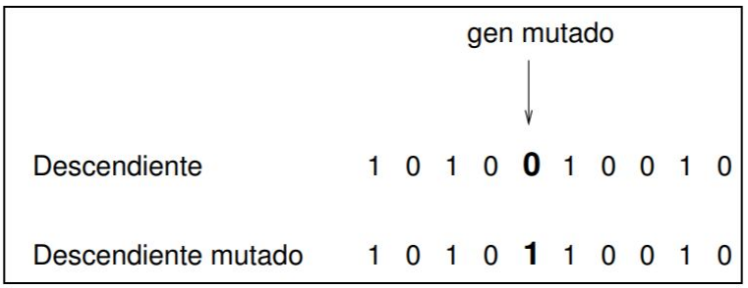


Figura 3: Algoritmo genético clásico

```
t ← 0
población(t) ← poblaciónInicial
EVALUAR(población(t))

while not (condición de terminación)
    t ← t + 1
    población(t) ← SELECCIONAR(población(t-1))
    población(t) ← CRUZAR(población(t))
    población(t) ← MUTAR(población(t))
    EVALUAR(población(t))

return población(t)
```

En la figura 3 podemos observar el algoritmo base para resolver un problema con algoritmos genéticos. La población evolucionará a lo largo de las generaciones de tal forma que la adaptación del mejor individuo irá incrementando hacia el óptimo global. Normalmente la condición de convergencia es cuando al menos el 95% de los individuos de la población comparten el mismo valor para dicho gen, este porcentaje puede ser adaptado al problema. Anteriormente se presentaron definiciones y técnicas básicas para resolver un problema con algoritmos genéticos, sin embargo, existen más técnicas y se puede utilizar la que se considere adecuada a lo que se está intentando resolver (UPV, 2003).

Aplicación de algoritmos genéticos en vehículos autónomos

Las empresas Waymo y DeepMind trabajan en conjunto entrenando algoritmos de redes neuronales para vehículos no tripulados. Ellos no lo hacen de una forma convencional, sino que utilizan un método similar al desarrollo evolutivo de los seres vivos. Para entender este mecanismo,

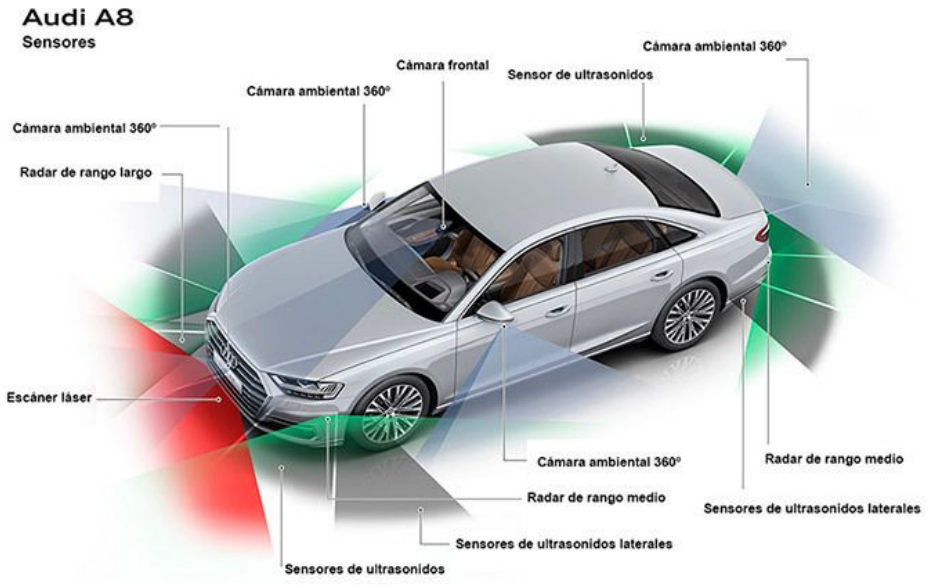
primero hay que mencionar que hay dos tipos de parámetros en los algoritmos de redes neuronales. Primero tenemos los parámetros discretos de una red neuronal, por ejemplo los pesos de cada neurona. Por otro lado, tenemos los hiperparámetros que son responsables de cómo se lleva a cabo el aprendizaje, por ejemplo la velocidad de aprendizaje con la que red aprende (Román, 2019).

Estas empresas en la forma que entrenan a sus redes es que inicialmente, los modelos comienzan el aprendizaje paralelo con un conjunto aleatorio de hiperparámetros. Los peores modelos son reemplazados con nuevas generaciones y luego se evalúan repetidamente como se explicó anteriormente en la teoría de algoritmos genéticos. Según la empresa este procedimiento ha mejorado el rendimiento de la conducción automatizada (Román, 2019).

Sensores utilizados para vehículos autónomos

Para fines prácticos, se tomará el vehículo Audi A8, y discutiremos todos los componentes que utiliza para llevar a cabo el manejo automatizado. Como podemos ver en la figura 4, este carro posee distintos sensores los cuales ayudan a calcular una imagen muy precisa del entorno fusionando los datos recogidos (ABC, 2017).

Figura 4: Representación gráfica de los sensores del Audi A8.



Se utilizan hasta doce sensores de ultrasonido, cuatro cámaras de visión periférica 360 grados, una cámara de vídeo 3D adicional en el borde superior del parabrisas, cuatro radares de medio alcance en las

esquinas del vehículo, un radar de largo alcance y una cámara infrarroja en la parte frontal, finalmente el nuevo escáner láser (ABC, 2017).

El sistema de radares se encarga de escanear lo que sucede por delante del coche, mientras que la cámara de vídeo 3D se encarga de distinguir objetos, por ejemplo reconocer las marcas de la carretera, otros vehículos o peatones. Los sensores de ultrasonido y el resto de cámaras supervisan el área completa que rodea al automóvil. Podemos apreciar que el carro posee demasiados sensores, estos sirven para validar la información por medio de comparaciones recíprocas. Puede sonar a redundancia innecesaria de datos, pero se necesita un reconocimiento detallado de los objetos, sin importar si son estáticos o dinámicos, así como si son o no metálicos (ABC, 2017).

Anexos y Resultados

Las pruebas realizadas consistieron en entrenar un carro con algoritmo genético y así obtener la cantidad de generaciones necesarias para hacer que este complete una vuelta a la pista sin salirse. Se varió la cantidad de sensores por cada set de pruebas; estos son visibles como líneas azules saliendo del carro propio.

Figura 5: Carro con 4 sensores.

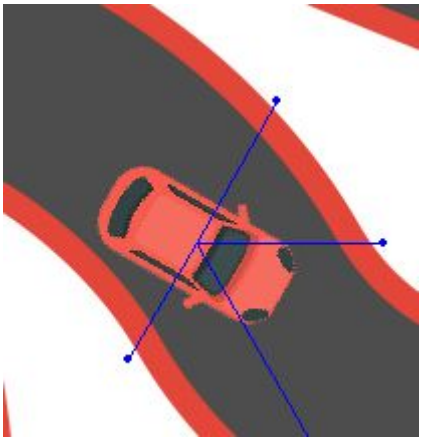


Figura 6: Carro con 5 sensores.

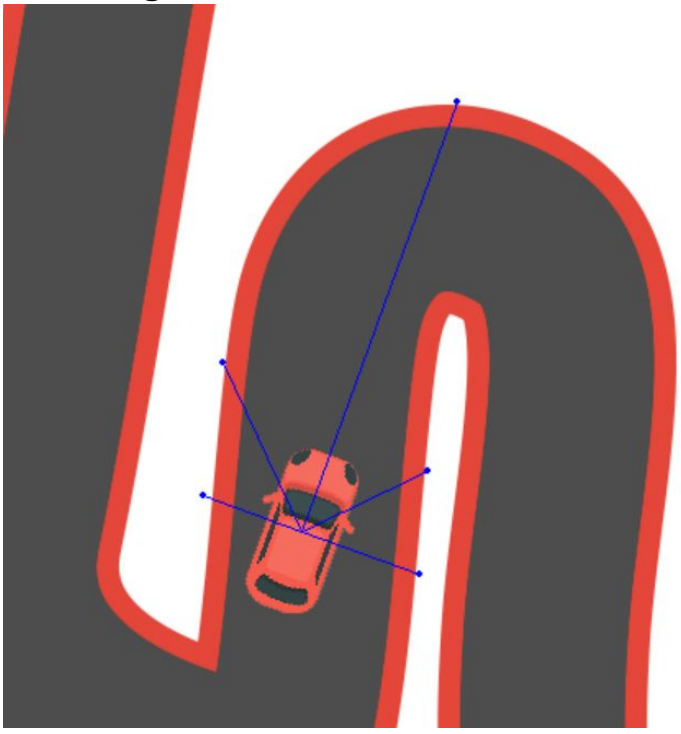


Figura 7: Carro con 9 sensores.

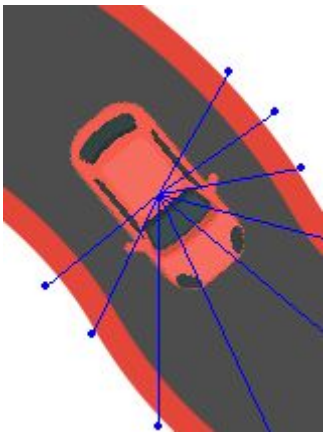


Figura 8: Carro con 11 sensores.

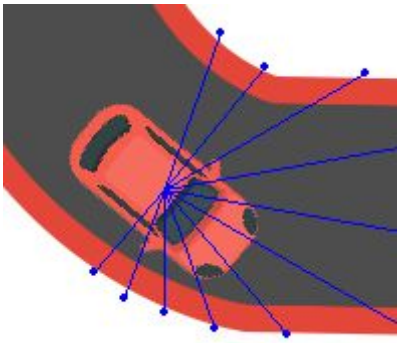


Tabla 1: resultados de simulación con 4 sensores.

Número de prueba	Generaciones necesarias para completar una vuelta
1	8
2	12
3	30
4	27
5	35
6	29
7	25
8	40

9	36
10	38

Tabla 2: resultados de simulación con 5 sensores.

Número de prueba	Generaciones necesarias para completar una vuelta
1	2
2	10
3	11
4	5
5	9
6	7
7	8
8	5
9	6
10	6

Tabla 3: resultados de simulación con 9 sensores.

Número de prueba	Generaciones necesarias para completar una vuelta
1	7
2	5
3	4
4	6
5	5
6	2
7	6

8	20
9	5
10	5

Tabla 4: resultados de simulación con 11 sensores.

Número de prueba	Generaciones necesarias para completar una vuelta
1	9
2	5
3	9
4	8
5	4
6	4
7	7
8	3
9	10
10	8

Figura 9: Comparación entre promedio de los resultados.

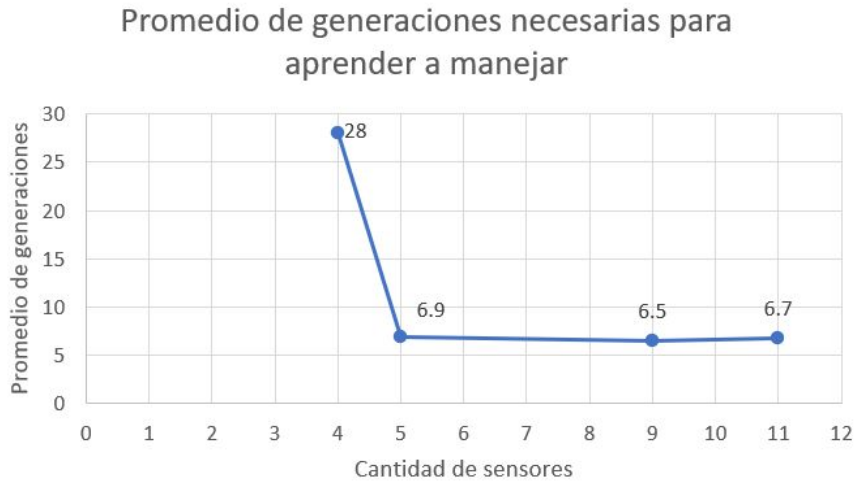
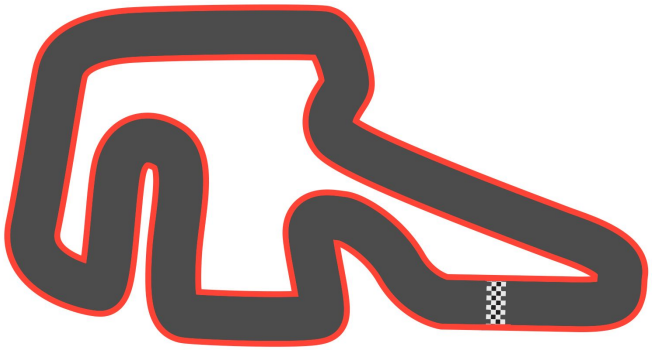


Figura 10. Pista utilizada para las pruebas.



Discusión de resultados

Luego de la parte investigativa, se percató que los carros autónomos hacen uso de varios sensores, por lo que resultó interesante simular estos sensores y comprender si la cantidad de ellos generaba un aprendizaje más rápido con ayuda de algoritmos genéticos. La hipótesis planteada es que la cantidad de sensores es inversamente proporcional a la cantidad de generaciones de un aprendizaje utilizando algoritmos genéticos.

Para llevar a cabo esta simulación se utilizó la librería de NEAT-Python, la cual es una implementación de NeuroEvolution of Augmenting Topologies. La red neuronal utilizada es bastante simple, consiste en x neuronas de entrada (representada como la cantidad de sensores de la prueba), en donde cada una de las entradas era la distancia entre el carro y la orilla de la pista; además de 2 neuronas de salida, las cuales representaban si el carro tenía que girar a la derecha o izquierda. Cabe mencionar que cada generación consistía de 30 carros y nuestra métrica de recompensa para cada individuo fue que tanto lograba avanzar por la pista sin salirse de esta.

En las figuras de la 5 a la 8 es posible notar cómo se distribuyeron los distintos sensores. Se realizaron 10 pruebas utilizando 3, 4, 5, 9 y 11 sensores. Esto con el fin de obtener el promedio de generaciones que era necesario para dar una vuelta completa a la pista. La pista utilizada se representa en la figura 10. Esta fue creada tomando en cuenta ser una pista difícil que cubriera tanto giros hacia la izquierda como a la derecha.

En la figura 9, se observa la comparación de promedios de cada una de las pruebas. No se incluye la prueba con 3 sensores debido a que no se logró entrenar un carro que pudiera completar la prueba, siempre tenía

un punto ciego en determinado punto de la pista. De igual forma, utilizando 4 sensores habían segmentos de la pista que se le dificultaron debido a los pocos sensores del carro, es por ello que tardó en promedio 28 generaciones en completar la prueba. Por otro lado, a partir de utilizar 5 sensores, los resultados fueron muy similares, estos resultados hacen que no se apruebe nuestra hipótesis ya que a partir de 5 sensores, no importa la cantidad de sensores la simulación se tarda aproximadamente la misma cantidad de generaciones.

Una de las razones por la cual la cantidad de generaciones necesarias no difieren mucho entre sensores, es porque al inicio de la prueba se da un peso aleatorio de los valores que el carro toma. Esto da lugar a resultados dependientes del azar y permite deducir que la cantidad de sensores óptima en uno de los carros para completar una vuelta es de 5.

Conclusiones

- La cantidad de sensores no es inversamente proporcional a la cantidad de generaciones necesarias para que el auto complete una vuelta.
- La cantidad de sensores óptima en un carro para evitar que se salga de la pista es de 5.
- Es posible entrenar un carro para que se maneje por sí solo con la implementación de algoritmos genéticos.

Literatura Citada

ABC. (2017). Así funcionan los sensores de última generación de un coche autónomo. Extraído de https://www.abc.es/motor/reportajes/abci-funcionan-sensores-ultima-generacion-coche-autonomo-201708140233_noticia.html?ref=https:%2F%2Fwww.google.com%2F

Román, V. (2019). Google entrenó sus vehículos autónomos con algoritmos evolutivos. Extraído de https://nmas1.org/news/2019/07/26/waymo-deepmind-evo-algo-autos

Berzal, F. (2014). Algoritmos Genéticos. Extraído de https://elvex.ugr.es/decsai/computational-intelligence/slides/G2%20Genetic%20Algorithms.pdf

Villegas, J. (2005). COMPARACIÓN DE TRES MODELO DE ALGORITMOS GENÉTICOS, UN ALGORITMO DE CONTEO Y UN ALGORITMO VORAZ A LA INFORMACIÓN DE 10 AÑOS DE LOS RENDIMIENTOS DE 40 EMISORAS DE LA BOLSA MEXICANA DE VALORES. Extraído de https://repositorio.tec.mx/bitstream/handle/11285/628278/33068000976018.pdf?sequence=1&isAllowed=y

UPV. (2003). ALGORITMOS GENÉTICOS. Extraído de http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/temageneticos.pdf