INTELIGENCIA ARTIFICIAL

Camila Andrea Borja Sánchez
Alejandro Sántibañez Sánchez
Leider Londoño
Daniel Cardozo

Noviembre 2019

Índice

Índice de figuras

1. Introducción

A lo largo de la historia y la evolución del pensamiento humano-matemático, los cambios y los avances se han hecho notorios permitiendo con ellos la creación de nuevo conocimiento científico que permite dar solución a problemas cotidianos y complejos que antes no se podían solucionar. La lógica difusa y lo sistemas difusos permiten dar un enfoque diferente a la resolución de problemas y situaciones de alta complejidad y que la lógica a priori no determina.

En el siguiente trabajo se tocarán las propiedades de los sitemas difusos, sus ecuaciones, operaciones, así como sus reglas; de igual manera de abordan los algoritmos genéticos, todo esto como medio para comprender el trasfondo de las grandes soluciones a problemas complejos.

2. Lógica Difusa

Quizás la aplicación en que la lógica Difusa ha conseguido un éxito mayor, y por ende un mayor número de seguidores, se encuentran en el Control Industrial. Aún cuando existen numerosas versiones de controladores que emplean lógica. Los algortimos de entrenamiento permiten ajustar el diseño del controlador para que tenga un comportamiento deseado, pero fuera de línea.

A continuación se presentan una serie de aplicaciones:

2.0.1. Controlador Difuso Adaptado

Presentamos aquí el ejemplo de un controlador aplicado al problema de llevar un hehículo con marcha hacia atrás a la velocidad constante para que busque una cierta línea recta y la siga. El problema ha sido desarrollado por Wang.

El vehículo está inicialmente ubicado a una distancia x de la línea, y formando un ángulo A con la recta normal a la línea. Se necesita diseñar un Controlador Difuso para que decida cuál es el ángulo de giro B que deben tener las ruedas del vehículo.

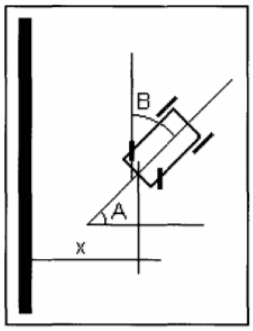


Figura 1: Controlador Difuso

Wang encuentra una Tabla de Parejas Entrada-Salida (como el cuadro 1) a partir de su propia experiencia como conductor, y con ella entrena un Controlador Difuso que tiene 5 valores lingüísticos para la distancia x, 7 para el ángulo A, y 7 para el ángulo de salida B. Los resultados los compara con los obtenidos por Nguyen y Widrow con redes neuronales a partir de la misma Tabla Entrada-Salida, y las trayectorias seguidas por el vehículo son semejantes. Wang también efectúa el entrenamiento con información numérica incompleta, es decir, considerando sólo una parte de la Tabla de entrenamiento; en estas condiciones el Controlador no es capaz de cumplir su objetivo. Sin embargo, al emplear Aplicaciones de la Lógica Difusa 7 algunas reglas If- Then extraidas de su experiencia como conductor, Wang obtiene resultados que son iguales a los del primer caso. Lo interesante de este último diseño es que allí se han combinado dos tipos de información de naturaleza diferente:

por una parte está la información numérica proveniente de la Tabla, y por otra parte está la información lingüística contenida en las reglas If-Then.

2.0.2. Identificador de Imágenes Aéreas

El problema aquí planteado es el siguiente: Se tienen tres imágenes aéreas de la misma zona; las imágenes se han captado empleando cámaras a blanco y negro, de tal forma que muestran en cada pixel un cierto nivel de gris (nivel de luminancia); las tres fotografías no son iguales, porque cada una de ellas se ha tomado anteponiendo a la cámara un filtro que sólo permite captar una franja de colores (unas ciertas longitudes de onda), y los filtros para cada fotografía han sido diferentes.

En estas imágenes se ha captado una área extensa con zonas que se pueden clasificar así:

- Zona de ríos.
- Zonas de construcciones humanas.

- Zonas dedicadas a la agricultura.
- Zonas Boscosas

El problema consiste en diseñar un algoritmo que, conociendo los niveles de luminancia para un cierto pixel en las tres fotografías, decida a cuál de las zonas anteriores corresponde ese pixel. No existe una única combinación de luminancias que identifique a cada una de las zonas (no hay un único «verde» en la zona de bosques), y por tanto no se conoce el conjunto de todas las combinaciones de luminancias posibles asociadas a cada zona.

2.0.3. Base de Datos Difusa

La Lógica Difusa busca desarrollar un conjunto de procedimientos para manejar la información precisa y/o vaga. Ahora bien, los Sistemas de Bases de Datos tienen por propósito, hablando en términos muy generales, la organización de la información; por tanto no es de sorprendente que se haya intentado incorporar las técnicas de Lógica Difusa en el diseño de Bases de Datos. Miyamoto y Umano distinguen dos tipos de técnicas difusas en las Bases de Datos: ·Bases de Datos Difusas. ·Técnicas Difusas para la recuperación de la información.

En la primera de estas técnicas el concepto de Conjunto Difuso se incorpora en la estructura misma de la Base de Datos, mientras que en la segunda se emplea

2.0.4. Aplicación de la lógica difusa en la toma de decisiones para la sostenibilidad del suelo

A partir del modelo Big6 para la solución de problemas de información, como modelo de competencias en el uso y manejo de la información, se desarrollaron los pasos siguientes para la realización del estudio.

- Establecer los componentes del problema a resolver y los indicadores asociados.
- Evaluar los tipos de suelos asociados en una tabla a los indicadores (síntomas) identificados asignándole valores entre 0 y1
- Contar las veces en que un tipo de suelo resulta evaluado igual o superior al que le sigue en una tabla, de esta operación de obtiene número enteros (2, 4,5, ...)
- Identificación de los suelos más afectados por los síntomas del mecanismo central del Síndrome de Sobre-Utilización del Suelo.

Número	Tipo de Suelo	Símbolo del Tipo de Suelo
1	Ferrítico Purpura Típico	S1
2	Ferraltico Pardo Rojizo x	S2
3	Ferrítico Purpura Concrecionario	S3
4	Ferrítico Purpura Laterizado	S4
5	Ferrítico Purpura Hidratado	S5
6	Ferralítico Rojo Típico	S6
7	Ferralítico Rojo Concrecionario	S7
8	Ferralítico Rojo Compactado	S8
9	Ferralítico Rojo Hidratado	S9
10	Ferralítico Rojo Poco Lixiviado	S10

Figura 2: Resultado

3. Conjunto Difuso

Matemáticamente, un conjunto es una colección de objetos que verifican alguna propiedad, de forma que un objeto o bien pertenece al conjunto, o no pertenece. Por ejemplo, supongamos que decimos que una persona es alta si su altura está por encima de 180cm, algo que puede ser representado gráficamente de la siguiente forma:

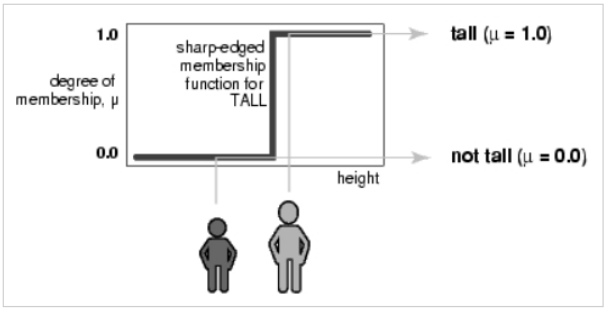


Figura 3: Ejemplo 1

La función anterior describe la pertenencia al conjunto de los altos, o estás dentro o estás fuera. Estas funciones nítidas funcionan muy bien con las operaciones matemáticas clásicas, pero no funcionan tan bien describiendo el mundo real. Por una parte, no hace distinción entre individuos que midan 181cm y los que miden 215cm, aunque hay una clara distinción entre ellos. El otro problema es la diferencia entre una persona que mida 180cm y otra de 181cm, apenas 1cm de diferencia entre ellos y el primero de ellos no está en el conjunto de los altos, y el segundo sí.

La aproximación de los conjuntos difusos al conjunto de los altos proporciona una representación mucho mejor sobre la propiedad ser alto de una persona. El conjunto se define por medio de una función continua que puede tomar valores

intermedios entre los extremos 0 y 1.

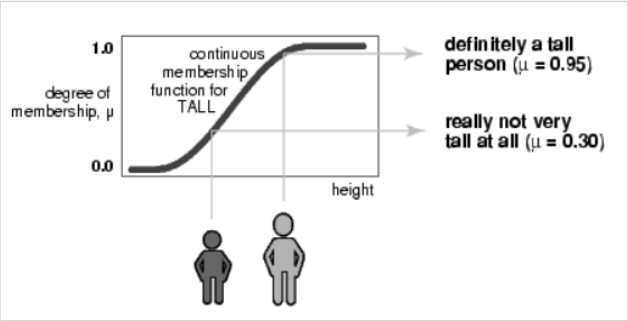


Figura 4: Ejemplo 2

En los conjuntos difusos relajamos la restricción de que la función de pertenencia valga ó 0 ó 1, y dejamos que tome valores en el intervalo [0,1]. A diferencia de los conjuntos nítidos, los conjuntos difusos sólo se pueden definir dando su función de pertenencia. Las funciones de pertenencia pueden emplearse de dos formas diferentes: – Para estimar grados de pertenencia a un conjunto Si nos dicen que una persona mide 170 cm, ¿en qué grado es alta?
Para expresar posibilidades en una situación en la que se dispone de información incompleta Si nos dicen que una persona es mediana, ¿cuál será su altura? En este caso la función de pertenencia puede interpretarse como una distribución de posibilidad que nos indica la preferencia sobre los valores que una variable de valor desconocido puede tomar.

4. Reglas Difusas

Una regla difusa es una expresión lingüística que refleja una causa y un efecto. Su calidad de difusa radica en el hecho de que emplea adjetivos imprecisos y relativos:

Si X pertenece a A, entonces Y pertenece a B Donde X y Y, son elementos; y, A y B, conjuntos difusos. Unos ejemplos de reglas difusas pueden ser:

- Si la temperatura es FRÍA, entonces el flujo de combustible es ALTO.
- Si el ciclo de trabajo es GRANDE, entonces la tensión es ALTA.
- Si la velocidad es ALTA, entonces la frenada es FUERTE.

En tales ejemplos, las variables “temperatura”, “ciclo de trabajo” y “velocidad” son partícipes en la causa, lo que puede verse como entradas al sistema y pudieran pertenecer a conjuntos difusos como “BAJO”, “GRANDE”, “ALTO”; respectivamente. Las variables “flujo de combustible”, “tensión” y “frenada”, son también elementos que serían “variables de salida” pertenecientes, en cierto grado, a conjuntos difusos “ALTO” o “FUERTE”

5. Propiedades de los conjuntos Difusos

Soporte: Es el conjunto de elementos cuyo grado grado de pertenencia es distinto de 0, se representa de la siguiente manera:
sop(A) = x — fA(x) ¿0, x E X

6. Funciones implementadas

6.1. Función de membresía

La función de membresía o de pertenencia de un conjunto nos indica el grado en que cada elemento de un universo dado, pertenece a dicho conjunto. Es decir, la función de pertenencia de un conjunto A sobre un universo X será de la forma: uA:X -¿[0,1], donde uA (x) = r si r es el grado en que x pertenece a A.

Si el conjunto es nítido, su función de pertenencia (función característica) tomará los valores en 0,1, mientras que si es borroso, los tomará en el intervalo [0,1]. Si uA(x) = 0 el elemento no pertenece al conjunto, si uA(x) = 1 el elemento sí pertenece totalmente al conjunto. Las funciones de pertenencia son una forma de representar gráficamente un conjunto borroso sobre un universo, entre estas se pueden enunciar la función Triangular y la función Trapezoidal que son las más utilizadas debido a su sencillez y simplicidad en sus cálculos

Las funciones de pertenencia se grafican sobre un conjunto de ordenadas "x" reflejadas en las ordenadas "y" en el intervalo de 0 a 1 como se ve en la siguiente imagen.

7. Función de Membresía

7.1. Función de saturación

También conocida como función de tipo hombro derecho o función L, es un caso especial de la función trapezoidal, viene definida por un límite a y soporte b, tal que a ¡b, los elementos menores que a serán iguales a a e irán hasta meno infinito, mientras que los elementos mayores que b serán iguales a b e irán hasta más infinito

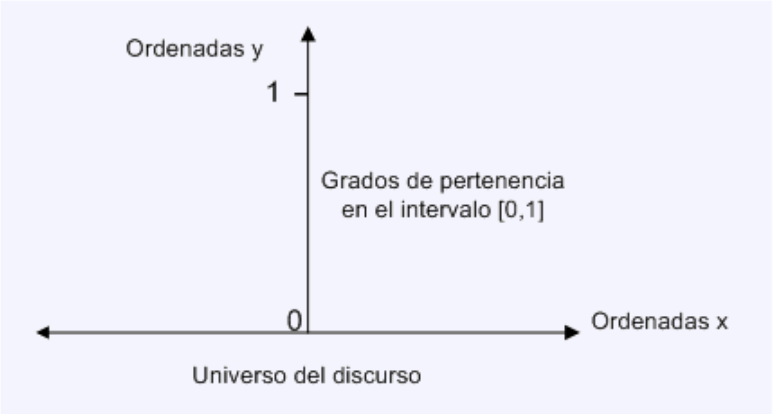


Figura 5: Función Membresía

8. Función de saturación

8.1. Implementación en python v.3.7

9. Implementación en python v.3.7

9.1. Función Hombro

También conocida como función hombro izquierdo o función R, como la anterior función esta es un caso especial de la función trapezoidal, viene definida por un límite d y soporte c, tal que c ¡d, los elementos mayores que d serán iguales a d e irán hasta más infinito, mientras que los elementos menores que c serán iguales a c e irán hasta menos infinito

10. Función Hombro

10.1. Implementación en python v.3.7

11. Implementaciónen python v.3.7

11.1. Función Triangular

Viene definida mediante el límite inferior a, el superior b y el valor modal m, tal que a¡m¡b. La función no tiene porqué ser simétrica.

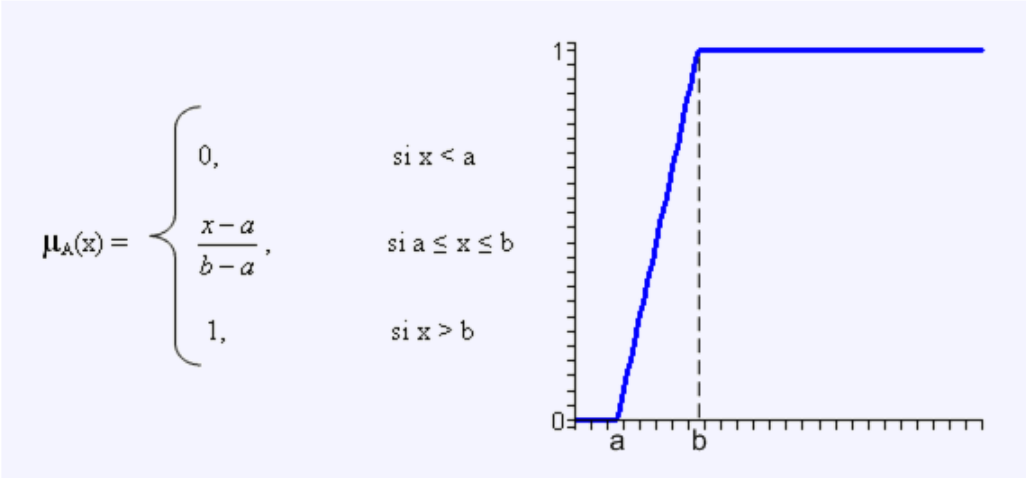


Figura 6: Función Saturación

12. Función Triangular

12.1. Implementación en python v.3.7

13. Implementaciónen python v.3.7

13.1. Función trapecio o pi

Definida por sus límites inferior a, superior d, y los límites de soporte inferior b y superior c, tal que a¡b¡c¡d. En este caso, si los valores de b y c son iguales, se obtiene una función triangular.

14. Función trapecio o pi

14.1. Implementación en python v.3.7

15. Implementaciónen python v.3.7

15.1. Función s o sigmoidal

Definida por sus límites inferior a, superior b y el valor m o punto de inflexión, tales que a¡m¡b. El crecimiento es más lento cuanto mayor sea la distancia a-b. Para el caso concreto de m=(a+b)/2, que es lo usual, se obtiene la siguiente gráfica.

```python
def funcion_saturacion(x, a, b):
    """
    Funcion Hombro derecho o saturacion

    Funcion que indica el grado de pertenencia de "x" a un conjunto de datos difuso, delimitado por los elementos "a" y "b" donde "a" indica
    el menor grado de pertenencia (0) y "b" el mayor grado de pertenencia (1) al conjunto, "a" debe ser menor que "b" (a < b)

    x = elemento a evaluar
    a = elemeto con menor grado de pertenencia (0), donde empieza recta con inclinacion positiva
    b = elemento con mayor grado de pertenencia (1), donde terminala la recta empezada en el punto "a"

    """
    if x <= a:
        result = 0
    elif a < x < b:
        result = (x-a)/(b-a)
    elif x >= b:
        result = 1

    return result
```

Figura 7: Implementación en python v.3.7

16. Función s o sigmoidal

16.1. Implementación en python v.3.7

17. Implementaciónen python v.3.7

- Altura de un conjunto difuso (height): Es el grado de pertenencia más grande de los elementos del conjunto: Altura(A) = Max h — h = fA(x), x E X
- Núcleo: Es el conjunto de elementos cuyo grado de pertenencia sea 1: Núcleo(A) = x E X / fA(x) = 1
- Conjunto difuso normal: Conjunto difuso que tiene una altura igual a 1. Altura(A) = 1

18. Algoritmos genéticos y sus aplicaciones

Es una técnica de búsqueda basada en la teoría de la evolución donde se intenta replicar el comportamiento biológico de la selección natural donde los individuos más aptos son seleccionados para la reproducción con el fin de producir descendencia de la próxima generación y la genética ya que son metodos adaptativos que pueden usarse para resolver problemas y tener una muy buena optimizacion.

Estos algoritmos son capaces de ir creando soluciones para problemas del mundo real, están diseñados para simular los procesos en los sistemas naturales necesarios para la evolución, especialmente aquellos que siguen los principios

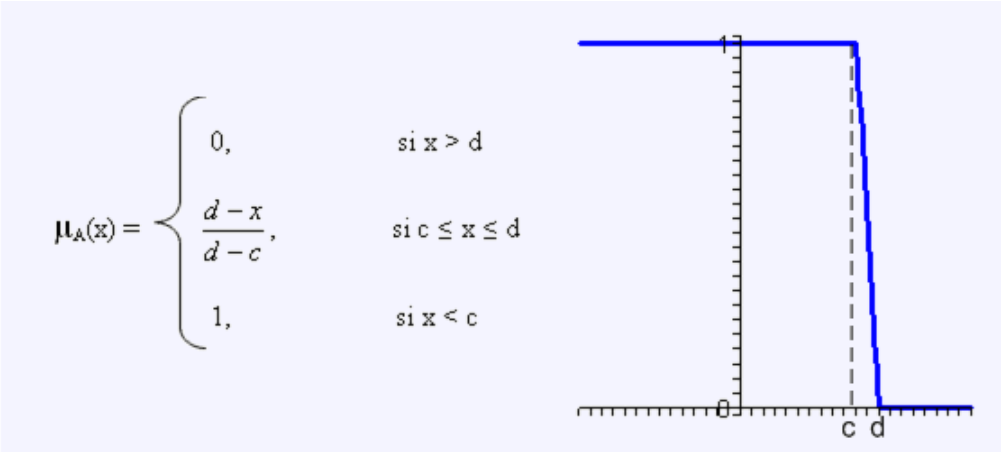


Figura 8: Función Hombro

establecidos por Charles Darwin de "la supervivencia del más apto", se refiere a la supervivencia de ciertos rasgos sobre otros que se reproducen en las siguientes generaciones, mientras que los otros rasgos tienden a desaparecer, como en la naturaleza, la competencia entre individuos por recursos escasos resulta en que las personas más aptas dominen a las más débiles.

En la naturaleza los individuos de una poblacion compiten entre sı en la busqueda de recursos tales como comida, agua y refugio. Incluso los miembros de una misma especie compiten a menudo en la busqueda de un compañero. Aquellos individuos que tienen mas exito en sobrevivir y en atraer compañeros tienen mayor probabilidad de generar un gran numero de descendientes. Por lo contrario individuos poco dotados produciran un menor numero de descendientes. Esto significa que los genes de los individuos mejor adaptados se van a propagarar en sucesivas generaciones hacia un numero de individuos en aumento.

Es decir los algoritmos geneticos trabajan con una poblacion de individuos, donde cada uno representa una solucion factible a un problema dado. A cada individuo se le asigna un valor o puntuacion, relacionado a dicha solucion.

El poder de los algoritmos geneticos proviene del hecho de que se trata de una tecnica robusta, y pueden tratar con exito una gran variedad de problemas provenientes de diferentes areas, incluyendo aquellos metodos en los que encuentran dificultades. Si bien no se garantiza que el algoritmo genetico encuentre una solucion optima del problema, existe evidencia que está basada en la experiencia y en la observación de los hechos que dan soluciones de un nivel aceptable, en un tiempo competitivo con el resto de algoritmos de optimizacion.

El campo de aplicación de los algoritmos geneticos es muy extenso y abarca areas como:

```
[ ] def funcion_hombro(x, a, b):
      """
        Funcion hombro izquierdo

        Funcion que indica el grado de pertenencia de "x" a un conjunto de datos difuso, delimitado por los elementos "a" y "b" donde "a" indica
        el grado de pertenencia mayor (1) y "b" el grado de pertenencia menor (0) al conjunto difuso, "a" debe ser menor que "b" (a < b)

        x = elemento a evaluar
        a = elemeto con mayor grado de pertenencia (1), donde empieza recta con inclinacion negativa
        b = elemento con menor grado de pertenencia (0), donde terminala recta empezada en punto "a"

      """
      if x <= a:
        result = 1
      elif a < x < b:
        result = (b-x)/(b-a)
      elif x >= b:
        result = 0

      return result
```

Figura 9: Implementación en python v.3.7

- Diseño de sistemas de dsitribucion de aguas.
- Diseño de topologias de circuitos impresos.
- Diseño de topologias de circuitos computacionales.
- investigacion de operaciones.
- Aprendizaje de comportamieno de robots.
- Aprendizaje de reglas de logica difusa.
- Optimizacion de estructuras moleculares.
- Diseño de controlador PID.
- Navegación robótica.
- Calibración de imágenes.
- Generación de datos de prueba para programas.
- Minería de datos usando.
- Optimización de red neural.

19. Historia de los algoritmos genéticos

Los primeros modelos de lo que hoy podríamos llamar algoritmos genéticos aparecieron a finales de los 50 y principios de los 60, programados en computadoras por biólogos evolutivos que buscaban realizar modelos de aspectos de

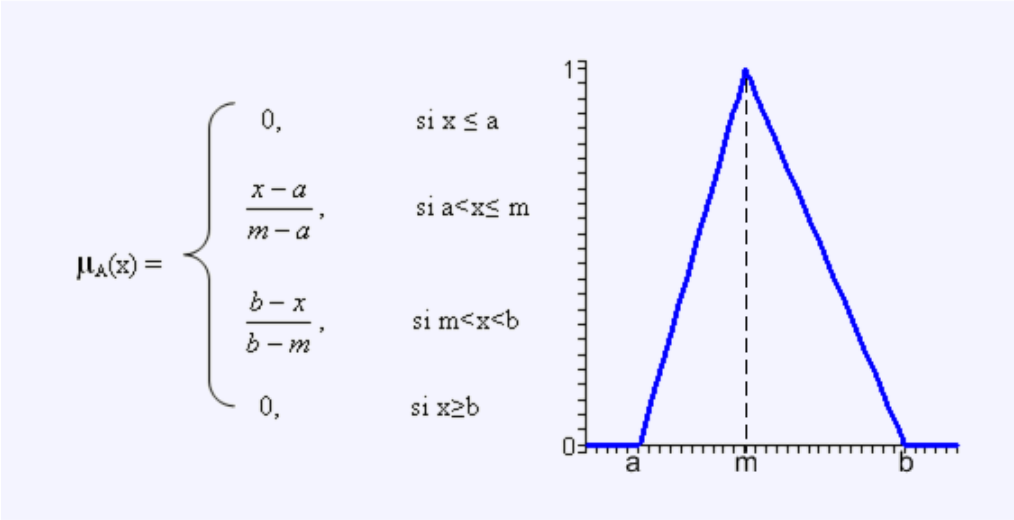


Figura 10: Función Triangular

la evolución natural, ninguno de ellos se le ocurrió que esta estrategia podría desarrollarse y aplicarse de manera más general a los problemas artificiales.

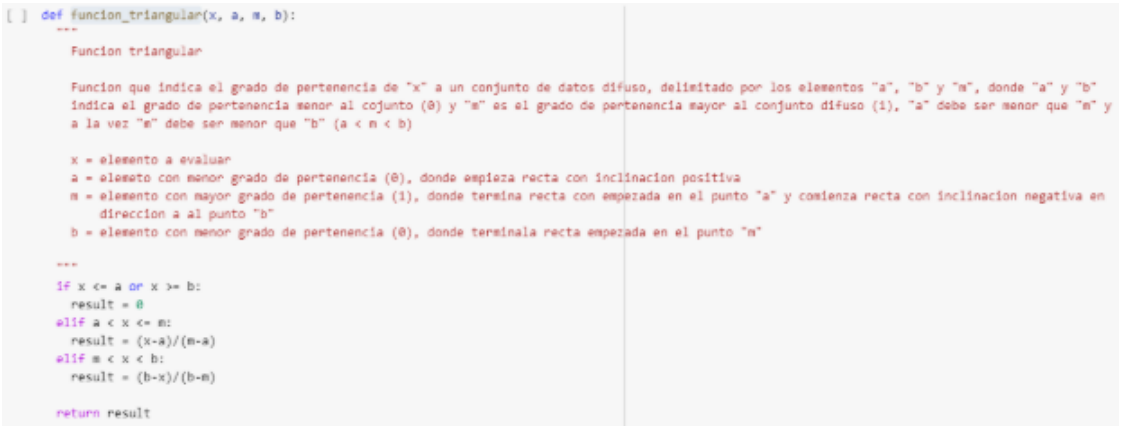
G.E.P. Box, G.J. Friedman, W.W. Bledsoe y H.J. Bremermann en 1962 desarrollaron algoritmos inspirados en la evolución para optimización de funciones y aprendizaje automático, pero sus trabajos acasionaron poca reacción.

En 1965 surgió un desarrollo, cuando Ingo Rechenberg de la Universidad Técnica de Berlín, introdujo una técnica que llamó estrategia evolutiva que se basaba en un padre que se mutaba para producir un descendiente, y se conservaba el mejor de los dos, convirtiéndose en el padre de la siguiente ronda de mutación.

El siguiente desarrollo vino en 1966, introdujeron en América una técnica que llamaron programación evolutiva, que consistia en las soluciones para los problemas se representaban como máquinas de estado finito sencillas, mutando aleatoriamente una de estas máquinas simuladas y conservando la mejor de las dos.

En los años 60 se establecieron las bases para desarrollos posteriores aunque lo curioso era que todo se lleva a cabo a base de interacciones locales entre individuos, entre éstos y lo que les rodea.

Fue en 1962 que sus ideas comenzaron a desarrollarse y a dar frutos en la Universidad de Michigan fue donde se crearon las ideas que más tarde se convertirían en los algoritmos genéticos.



```
[ ] def funcion_triangular(x, a, m, b):
      """
        Funcion triangular

        Funcion que indica el grado de pertenencia de "x" a un conjunto de datos difuso, delimitado por los elementos "a", "b" y "m", donde "a" y "b"
        indica el grado de pertenencia menor al cojunto (0) y "m" es el grado de pertenencia mayor al conjunto difuso (1), "a" debe ser menor que "m" y
        a la vez "m" debe ser menor que "b" (a < m < b)

        x = elemento a evaluar
        a = elemeto con menor grado de pertenencia (0), donde empieza recta con inclinacion positiva
        m = elemento con mayor grado de pertenencia (1), donde termina recta con empezada en el punto "a" y comienza recta con inclinacion negativa en
            direccion a al punto "b"
        b = elemento con menor grado de pertenencia (0), donde terminala recta empezada en el punto "m"

      """
      if x <= a or x >= b:
        result = 0
      elif a < x <= m:
        result = (x-a)/(m-a)
      elif m < x < b:
        result = (b-x)/(b-m)

      return result
```

Figura 11: Implementación en python v.3.7

20. ¿Por qué usar algoritmos genéticos, que ventajas y desventajas tiene?

Los algoritmos genéticos son técnicas factibles a ser aplicadas en la resolución de problemas de planificación por su flexibilidad, solo utilizan información de la función objetivo, sin necesidad de conocimiento previo como derivadas u otro conocimiento auxiliar. Además, permiten la resolución de problemas multiobjetivos de forma simultánea.

En general la aplicación de los algoritmos genéticos en el problema que se ejecuta simultáneamente arroja buenos resultados, pero se recomienda seguir profundizando en la solución a este por su complejidad y su aplicabilidad.

20.1. Ventajas

- No requieren conocimientos específicos del problema para llevar a cabo la búsqueda.
- Usan operadores aleatorios en vez de operadores deterministicos, lo que hace que la convergencia de la tecnica varie con respecto al tiempo.
- Operan de forma simultanea con varias solcuones tomando informacion de varios puntos del espacio de busqueda como guia.
- Resultan menos afectados por los maximos locales que las tecnicas de busqueda tradicionales.

20.2. Desventajas

- El lenguaje que se debe utilizar debe tener la capacidad de tolerar cambios aleatorios; que no lleguen a producir resultados sin sentido o errores

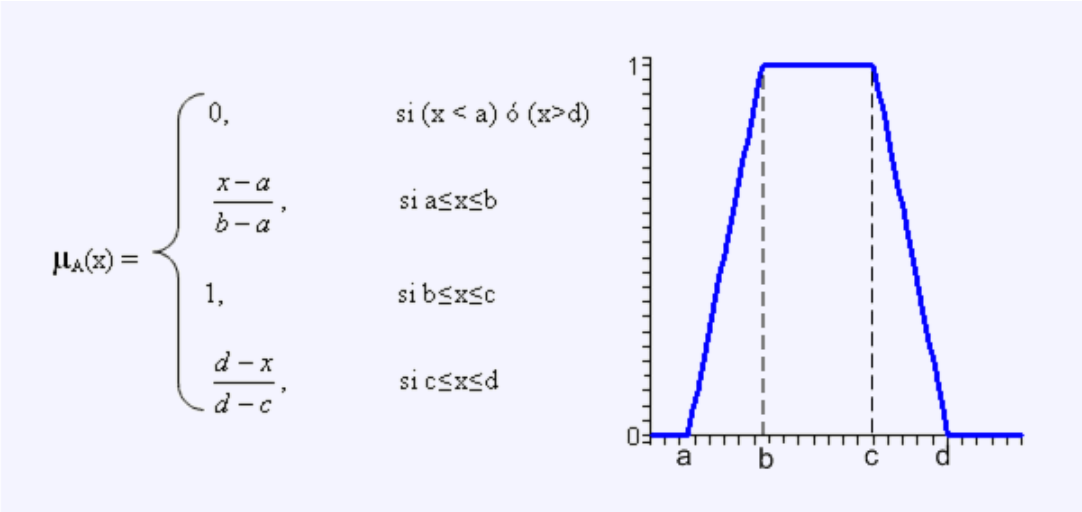


Figura 12: Función trapecio o pi

fatales.

- Puede demorarse bastante en converger, esto depende de cierto modo en los parámetros que se estén utilizando.
- Pueden converger prematuramente debido a una serie de problemas.

Son métodos de adaptación que pueden ser utilizados para implementar búsquedas y solucionar problemas de optimización:

- Problema del viajante
- Mastermind
- 8-puzzle

21. Operaciones Conjuntos Difusos

Las operaciones básicas entre conjuntos difusos son las siguientes:

```python
def funcion_trapecio(x, a, b, c, d):
    """
    Funcion trapecio o pi

    Funcion que indica el grado de pertenencia de "X" a un conjunto de datos difuso, delimitado por los elementos "a", "b", "c" y "d"; donde "a"
    y "b" indica el grado de pertenencia menor al cojunto difuso (0) y "b" y "c" son el grado de pertenencia mayor al conjunto difuso (1),
     "a < b < c < d".

    x = elemento a evaluar
    a = elemeto con menor grado de pertenencia (0), donde empieza la inclinacion positiva de la recta
    b = elemento con mayor grado de pertenencia (1), donde termina la recta con inclinacion positiva empezada en el punto "a" y comienzo de la
        recta paralela al eje de los elementos
    c = elemento con mayor grado de pertenencia (1), donde termina la recta paralela al eje de los elementos empezada en el punto "b"
    d = elemento con menor grado de pertenencia (0), donde terminala inclinacion negativa de la recta empezada en el punto "c"

    """

    if x <= a or x >= d:
        result = 0
    elif a < x < b:
        result = (x-a) / (b-a)
    elif b <= x <= c:
        result = 1
    elif c < x < d:
        result = (d-x) / (d-c)

    return result
```

Figura 13: Implementación en python v.3.7

$$\mu_A(x) = \begin{cases} 0, & \text{si } x \le a \\ 2\left[\dfrac{x-a}{b-a}\right]^2, & \text{si } a < x \le m \\ 1-2\left[\dfrac{x-b}{b-a}\right]^2, & \text{si } m < x < b \\ 1, & \text{si } x \ge b \end{cases}$$

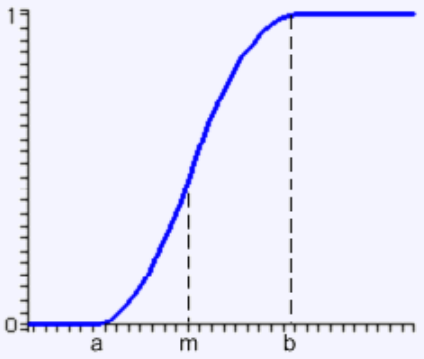


Figura 14: Función s o sigmoidal

```python
def funcion_sigmoidal(x, a, b, m):
    """
    Funcion sigmoidal o s

    Funcion que indica el grado de pertenencia de "x" a un conjunto de datos difuso, delimitado por los elementos "a" y "b", "m" es el punto de
    inflexión; donde "a" indica el grado de menor pertenencia al cojunto difuso (0), "b" el grado de mayor pertenencia (1), "m" el punto de
    inflexión calculado normalmente por "m = (a+b)/2", "a" debe ser menor que "m" y al mismo tiempo que "m" debe ser menor que "d" (a < m < d)

    x = elemento a evaluar
    a = elemeto con menor grado de pertenencia (0)
    m = punto de inflexión
    d = elemento con mayor grado de pertenencia (1)

    """
    if x <= a:
        result = 0
    elif a < x <= m:
        result = 2 * ((x-a)/(b-a))**2
    elif m < x < b:
        result = 1 - 2 * ((x-b)/(b-a))**2
    elif x >= b:
        result = 1

    return result
```

Figura 15: Implementación en python v.3.7

El conjunto complementario $\overline{A}$ de un conjunto difuso A es aquel cuya función característica viene definida por:

$$\mu_{\overline{A}}(x) = 1 - \mu_A(x)$$

La unión de dos conjuntos difusos A y B es un conjunto difuso $A \cup B$ en U cuya función de pertenencia es:

$$\mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x))$$

La intersección de dos conjuntos difusos A y B es un conjunto difuso $A \cap B$ en U con función característica:

$$\mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x))$$

Figura 16: Operaciones Básicas