Estudiantes
Julian Leonardo Robles Cabanzo
Diego Fernando Malagon Saenz

4. Ejercicios y Problemas

Ejercicios propuestos

1. **Descargue MEPX, https://www.mepx.org/, estúdielo y corra uno de los ejemplos que trae.**

   Multi Expression Programming (MEP) es una técnica evolutiva desarrollada para la generación automática de programas informáticos. A diferencia de otros enfoques como la Programación Genética (GP), MEP se diferencia por introducir la codificación de múltiples soluciones dentro de un solo cromosoma, permitiendo explorar más eficientemente el espacio de búsqueda sin penalizar el tiempo de ejecución ni los recursos computacionales.
   El funcionamiento de MEP se basa principalmente en representar individuos como secuencias lineales de genes, donde cada gen puede ser una variable o una función. Cada cromosoma codifica varias expresiones matemáticas y durante el proceso evolutivo se evalúan todas para seleccionar la mejor según una función de aptitud. Esta estructura es importante pues facilita la reutilización de subexpresiones y mejora la eficiencia del algoritmo, haciendo que MEP demuestre ser competitivo frente a otros métodos evolutivos como GEP, LGP y CGP, especialmente en tareas de regresión simbólica y clasificación.
   Ahora bien, MEPX es el software oficial que implementa esta técnica, compatible con todos los sistemas y con interfaz gráfica amigable, diseñada para resolver problemas de regresión simbólica, clasificación binaria y multiclase, y series temporales, esta nos permite cargar datos tabulares, configurar parámetros evolutivos, visualizar resultados y exportar fórmulas generadas en formatos como Excel o Python. MEPX también incluye estrategias avanzadas para clasificación multiclase y predicción de series temporales con múltiples salidas.
   En cuanto a su historia, MEP fue propuesto por Mihai Oltean y D. Dumitrescu en 2002 como una evolución de los paradigmas existentes en programación genética. Desde entonces, ha sido objeto de múltiples publicaciones científicas y mejoras continuas. MEPX ha evolucionado con actualizaciones frecuentes que incluyen mejoras en la interfaz, nuevas funciones matemáticas, soporte para exportación de fórmulas y optimización del rendimiento.
   Las aplicaciones de MEPX son amplias: análisis de datos, descubrimiento de fórmulas matemáticas, diseño de estrategias de juego, generación de heurísticas, modelado de sistemas dinámicos, y más. Su capacidad para manejar múltiples soluciones por cromosoma lo hace especialmente útil en problemas complejos donde se requiere una exploración profunda del espacio de soluciones.
   Como parte de trabajo investigativo se opto por ejecutar uno de los ejemplos del programa, sobre clasificación binaria que llevaba el nombre de çancer1 bat class labels changed", esto se puede ver en la figura 1.

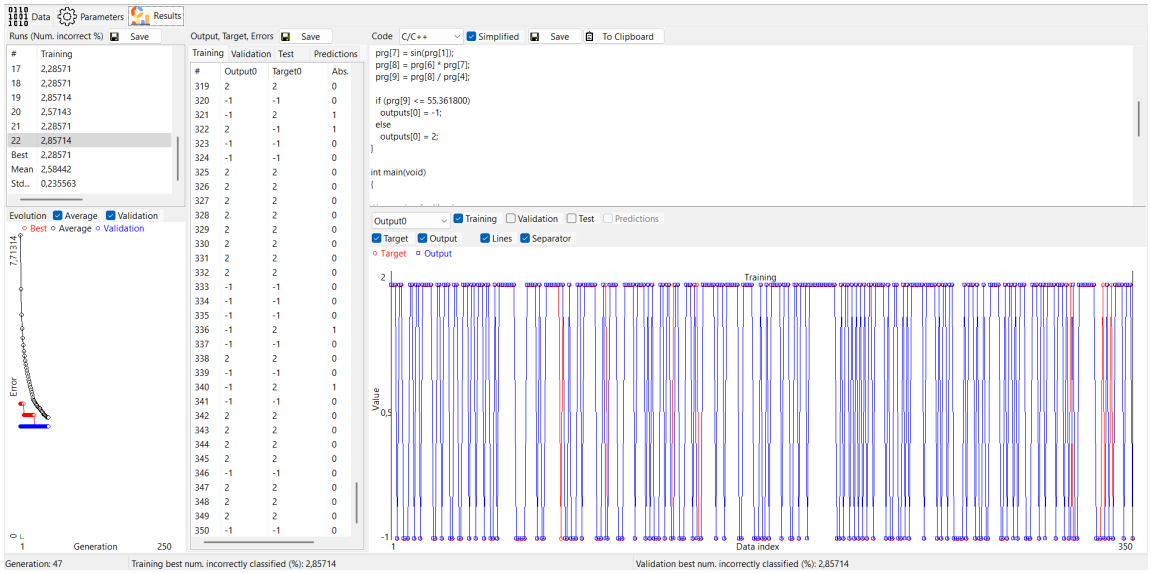

Figura 1: Resultados de la prueva de ejemplo en MEPX.

El progama se tardó bastante tiempo y se logran identificar hasta 22 iteraciones, pero logrando un muy bajo margen de error, como se puede ver en los datos obtenidos. El resultado final del software es un código implementa un clasificador binario mediante una expresión matemática no lineal. La lógica principal reside en la función mepx(), que procesa 9 características de entrada (x[0] a x[8]) y produce una clasificación binaria (-1 o 2).

Secuencia de operaciones:

- prg[0] = x[6] → Captura característica 6
- prg[1] = log(prg[0]) → Logaritmo natural de x[6]
- prg[2] = x[5] → Captura característica 5
- prg[5] = prg[2] * prg[3] → Multiplicación (x[5] * x[2])
- prg[6] = prg[1] / prg[5] → División [log(x[6]) / (x[5]*x[2])]
- prg[7] = sin(prg[1]) → Seno del logaritmo
- prg[8] = prg[6] * prg[7] → Producto de resultados intermedios
- prg[9] = prg[8] / prg[4] → Resultado final normalizado.
- Finalmente realiza la clasificación poniendo como umbral 55.36, si es menor o igual, entonces la salida es -1, y si es mayor entonces la salida es 2.

La ejecución del ejemplo de clasificación binaria en MEPX demuestra que el software cumple eficazmente su propósito de generar fórmulas evolutivas funcionales a partir de datos tabulados. El código obtenido revela una capacidad sólida para identificar patrones no lineales mediante operaciones matemáticas complejas, con una lógica fácilmente integrable en sistemas embebidos. Aunque la interpretación de la fórmula puede ser abstracta, su rendimiento como clasificador automático basado en umbrales evidencia el potencial del enfoque evolutivo en tareas de decisión. Esta prueba valida el uso de MEPX como herramienta poderosa para resolver problemas de clasificación con independencia de estructuras predefinidas

2. **Suponga que tiene un robot que le entrega galletas al grupo de ingenieros de diseño de robots. Programe por PG el recorrido del robot, teniendo en cuenta que cada vez que un ingeniero recibe una galleta gana puntos. Los ingenieros están distribuidos en una sala cuadrada. Defina, conjunto de terminales, conjunto de funciones y función de aptitud.**

   Para esto entonces, se empezo con la implementación de la libreria deap, una de las más utilizadas para programación de algoritmos evolutivos. Inspirado en la evolución biologica, esta tecnica crea algoritmos evolutivos (selección, cruce, mutación) para generar una población de ïndividuos"(programas) los cuales cada uno se trata de un árbol sintáctico que representa código (ej: funciones, operadores matemáticos, variables), dicho esto los mejores programas sobreviven y se reproducen optimizando una función objetivo la cual en este caso es la programación de un robot repartidor de galletas.

   Sin embargo este primer intento no obtuvo los resultado deseados, el comportamiento del robot era siempre lineal y por lo tanto no cumplía con las expectativas, se intentó trabajar con diferentes combinaciones de generaciones y poblaciones, el código fue planteado de diferente maneras, sin embargo no se tuvo éxito en este desarrollo.

   Fue aquí cuando se requirió ayuda de la inteligencia artificial claude, con la cual se planteó un enfoque diferente sin el uso de la librería deap, en su lugar se planteó un código estructurado orientado a objetos donde se definía como objeto al robot, los ingenieros que representaban el objetivo y a la programación genética como tal. A partir de esta base se planteó un código basado en los intentos anteriores, con una población de 50 y una cantidad de 30 generaciones.

   Está versión del código fue contruido con:

   1. Conjunto de Terminales:

      - movernorte, moversur, movereste, moveroeste: Movimientos básicos.
      - entregargalleta: Entrega galleta si hay un ingeniero cerca
      - buscaringenierocercano: Se mueve hacia el ingeniero más próximo
      - ircentro: Regresa al centro de la sala

   2. Conjunto de Funciones:

      - secuencia: Ejecuta dos acciones consecutivamente
      - sihaygalletas: Condicional basado en galletas disponibles
      - repetir: Repite una acción varias veces
      - siingenierocerca: Condicional basado en proximidad.

   3. Función de Aptitud que se define con la siguiente formula:

$$Aptitud = (galletasentregadas100) + (cobertura2) + (50 - distanciaexcesiva) \tag{1}$$

   En cuanto a la representación del espacio se definió una matriz de 10x10, se planteó un total de 8 ingenieros y se definió un sistema de recompensas para incitar al robot a encontrar a los ingenieros y a entregarles su galleta. El primer resultado de este código final se encuentra en la figura 2 donde se ve el esquematico del cuarto y del recorrido del robot y en la figura 5 donde se puede ver el diagrama de la evolución del programa.

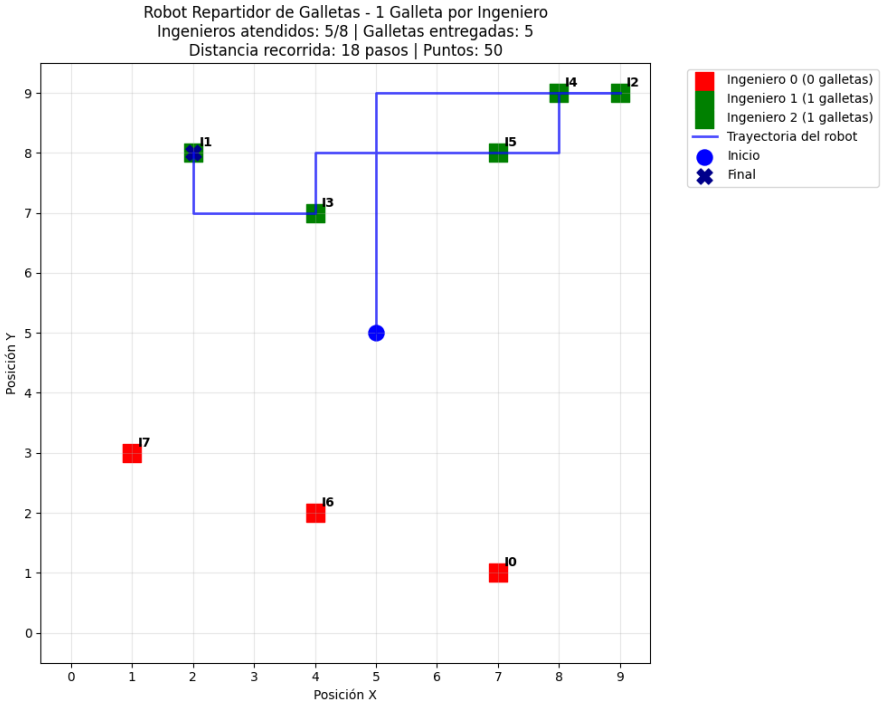

Figura 2: Representación grafica del experimento con población de 50 y 30 generaciones.

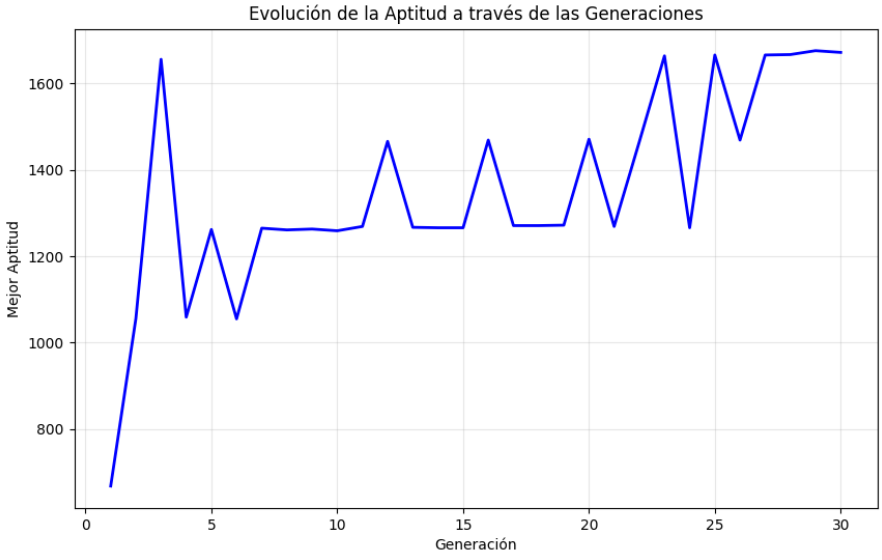

Figura 3: Evolución de la aptitud en función de las generaciones con población de 50 y 30 generaciones..

Al no conseguir los mejores resultado se aumento la población a 300 y las generaciones a 50 obteniendo ahora

si unos resutados mucho más aceptables, pues en la figura 4 se puede ver como ya puede encontrar a todos los ingenieros y en la figura 5 se observa una curva de evolución en función de la aptitud mucho más coherente.

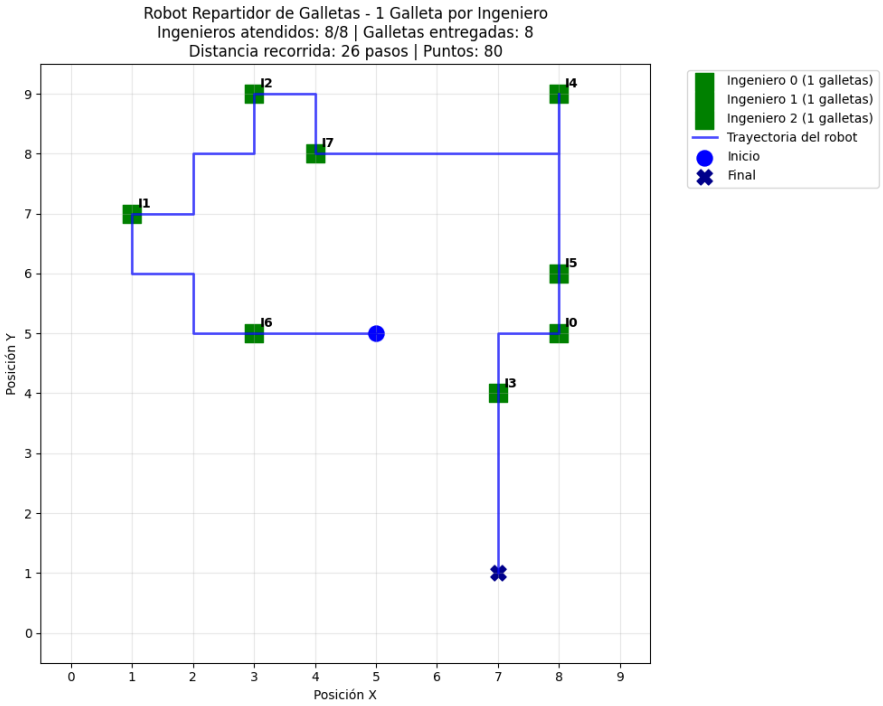

Figura 4: Representación grafica del experimento con población de 300 y 50 generaciones.

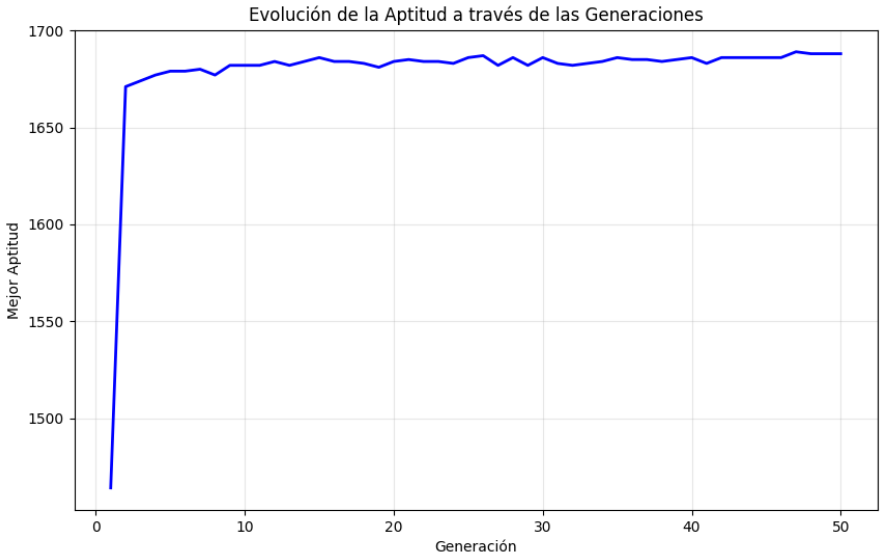

Figura 5: Evolución de la aptitud en función de las generaciones con población de 300 y 50 generaciones..

Como conclusión se puede afirmar que la programación genética es una herramienta poderosisima y súper interesante, llegando a ser lo suficientemente competitiva respecto a otra técnicas de IA, sin embargo requiere un gran estudio y profundización para poder implementarlo de manera correcta.