Computación Evolutiva

Una breve introducción

Luis Alfredo Alvarado Rodríguez

Distelsa

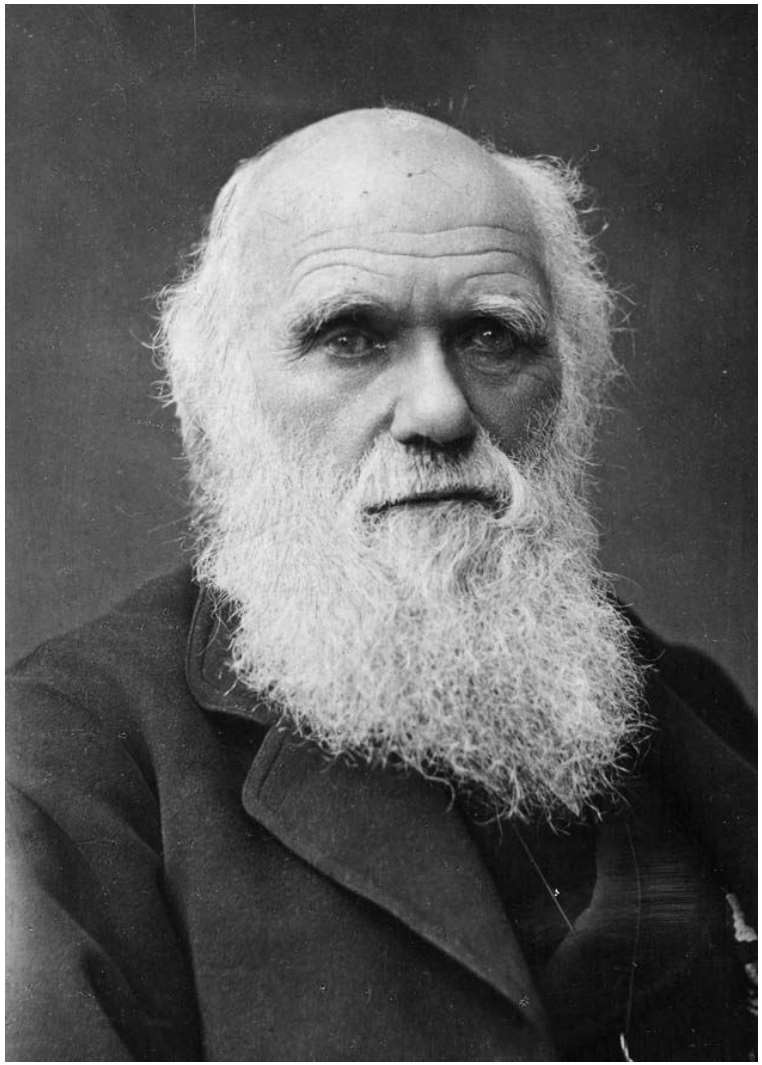
luis.alvarado@distelsa.com.gt

Weekly
October 24, 2024

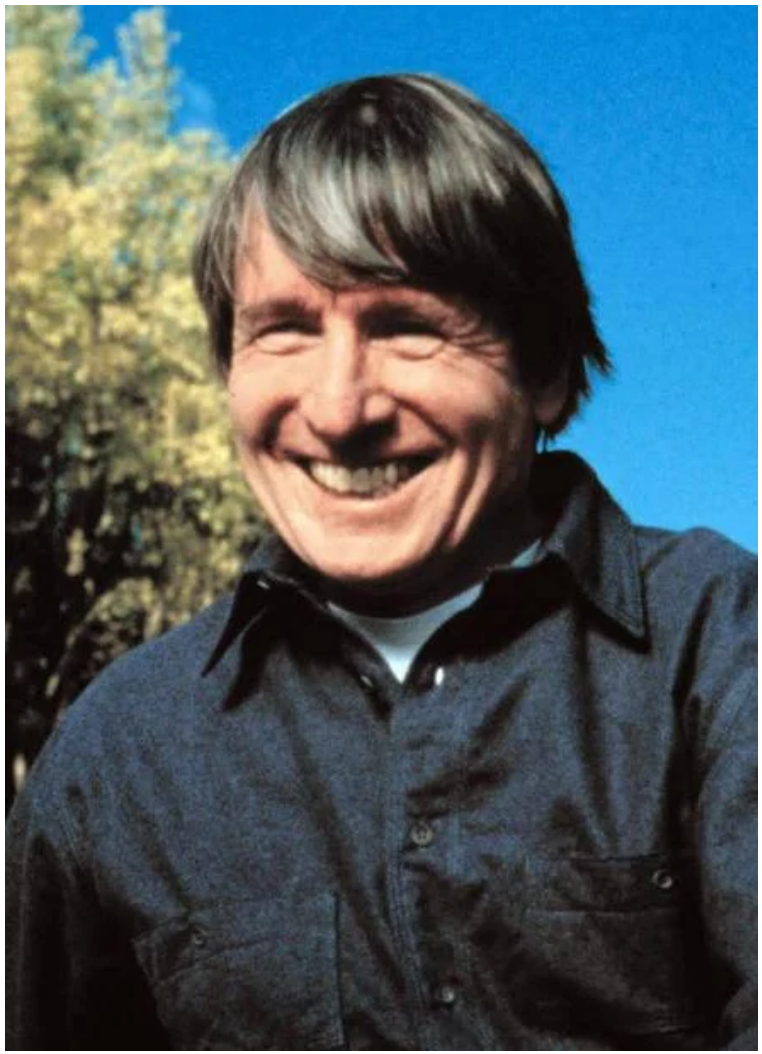
Descripción general de la presentación

La computación evolutiva es un área de las ciencias de la computación que se enfoca al estudio de las propiedades de una serie de metaheurísticas estocásticas (a las cuales se les denomina "Algoritmos Evolutivos") inspiradas en la teoría de la evolución de las especies formulada por Charles Darwin, según la cual los individuos "más aptos" a su ambiente tienen una mayor probabilidad de sobrevivir.
— ***Computación Evolutiva, amexcomp.***

La computación evolutiva comenzó a desarrollarse en la década de 1960 como una rama de la inteligencia artificial y la investigación operativa, pero sus raíces conceptuales se basan en la teoría de la evolución de **Charles Darwin**.



John Holland es uno de los pioneros más importantes en este campo. En los años 70, desarrolló el concepto de Algoritmos Genéticos (AG), uno de los primeros y más conocidos enfoques de la computación evolutiva. Los algoritmos genéticos imitan el proceso biológico de la evolución, utilizando procesos de selección, cruzamiento y mutación para mejorar iterativamente las soluciones a un problema.



Desde la introducción de los algoritmos genéticos, el campo ha crecido para incluir una variedad de enfoques evolutivos, como las **estrategias evolutivas (ES)** y la **programación genética (GP)**. Estos enfoques han permitido aplicar los principios de la evolución a problemas de diseño, simulación y optimización en diversas disciplinas, como la biología, la ingeniería y la economía.

Casos de Uso de la Computación Evolutiva

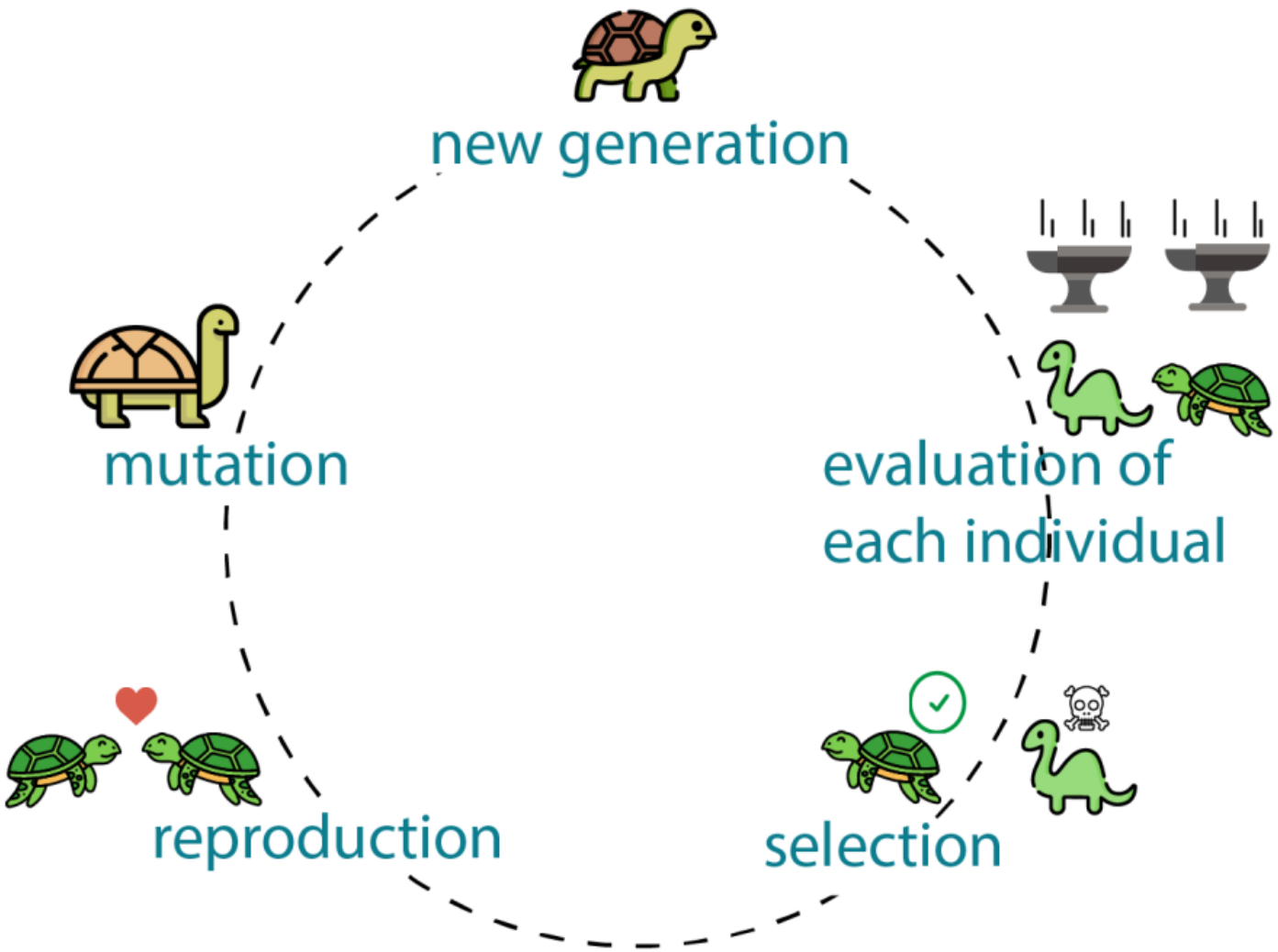
- **Optimización de ingeniería**: Diseño de circuitos, rutas óptimas, diseño mecánico, etc.
- **Inteligencia artificial en juegos**: Ejemplo de criaturas evolutivas en simuladores de videojuegos.
- **Machine Learning**: Evolución de hiperparámetros en redes neuronales y otros modelos.
- **Investigación científica**: Aplicación en biología computacional (por ejemplo, simulación de evolución de poblaciones).
- **Economía y finanzas**: Uso en la optimización de carteras de inversión o la simulación de mercados.

Los **algoritmos evolutivos (AEs)** son modelos computacionales que simulan procesos evolutivos para resolver problemas de optimización y búsqueda. Se basan en los mecanismos de la evolución biológica, como la selección natural, el cruzamiento y la mutación, y se utilizan cuando no existe un método determinístico que pueda encontrar la solución óptima de manera eficiente. Los AEs son útiles en problemas donde el espacio de soluciones es vasto y las soluciones óptimas no están claramente definidas.

- **Selección natural**: Proceso en el cual los mejores individuos (soluciones) de una población tienen mayor probabilidad de reproducirse, transmitiendo sus características (genes) a la siguiente generación.
- **Mutación**: Introducción de cambios aleatorios en los individuos para explorar nuevas soluciones en el espacio de búsqueda.
- **Cruzamiento (Recombinación)**: Mezcla de las características de dos individuos para generar nuevos individuos con características combinadas, con el objetivo de explorar mejor el espacio de soluciones.

Fundamentos Teóricos de la Computación Evolutiva

Inspiración biológica



1. **Algoritmos Genéticos (AGs):**
   Los **algoritmos genéticos (AGs)** son quizás el tipo más conocido de algoritmos evolutivos. Inspirados en los mecanismos de la genética biológica, se enfocan en el cruce y la mutación de genes (representación binaria o de otro tipo) para generar nuevas soluciones.
   - Usan una población de cromosomas (individuos) que se combinan y mutan para buscar mejores soluciones.
   - Aplicaciones: Optimización de rutas, problemas de diseño, inteligencia artificial en videojuegos.

2. **Estrategias Evolutivas (ESs):**
   Las **estrategias evolutivas (ESs)**, desarrolladas por Rechenberg y Schwefel, se centran en la evolución de parámetros continuos y son especialmente útiles para resolver problemas de optimización numérica. Las ESs se diferencian de los algoritmos genéticos en que hacen más hincapié en la mutación que en el cruzamiento.
   - Utilizan la mutación como principal operador de variación, y la recombinación (cruzamiento) se usa de manera opcional o limitada.
   - Aplicaciones: Optimización de parámetros en sistemas mecánicos, diseño de estructuras, calibración de sistemas de control.

3. **Programación Genética (GP):**
   La **programación genética (GP)** es un tipo de algoritmo evolutivo en el que los individuos son programas de computadora. Los programas se evalúan en función de su capacidad para resolver un problema determinado, y luego evolucionan a través de cruces y mutaciones para generar mejores programas.
   - En lugar de soluciones representadas como vectores o cadenas de bits, en GP los individuos son estructuras de programas, como árboles de expresiones.
   - Aplicaciones: Automatización de código, diseño de algoritmos, aprendizaje automático.

1. **Inicialización: Generación de la población inicial**
   El primer paso en un algoritmo evolutivo es generar una población inicial de individuos. Cada uno de estos individuos representa una posible solución al problema, y puede codificarse como una cadena de datos (por ejemplo, una secuencia binaria, un vector de números reales, o incluso un programa en el caso de la programación genética).
   - **Población inicial**: Puede ser generada de manera aleatoria o usando algún conocimiento previo del problema para mejorar las soluciones iniciales.
   - **Tamaño de la población**: El número de individuos en la población varía según el problema y afecta el rendimiento del algoritmo. Poblaciones más grandes permiten explorar más el espacio de búsqueda, pero aumentan el costo computacional.

2. **Evaluación: Aplicación de la función de fitness**
   Una vez generada la población inicial, se evalúa a cada individuo aplicando la **función de fitness**. Esta función mide la calidad de cada solución, determinando qué tan ”buena” o ”adecuada” es en comparación con otras soluciones. Los individuos con un mejor valor de fitness tendrán mayor probabilidad de ser seleccionados para la siguiente generación.
   - **Función de fitness**: Dependiendo del problema, puede ser una medida directa (por ejemplo, la distancia mínima en un problema de rutas) o una evaluación basada en una simulación.
   - **Evaluación cuantitativa**: Asigna un valor numérico que refleja el rendimiento o la calidad de la solución.

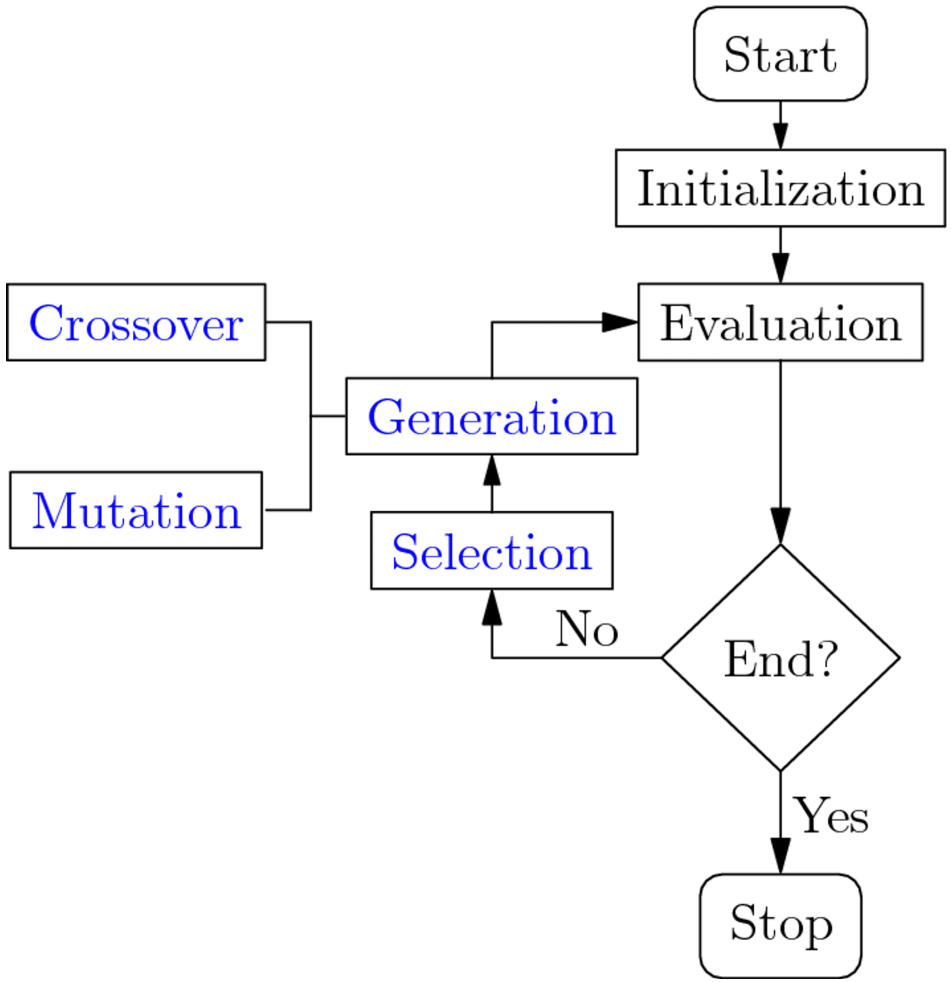
3. **Selección: Supervivencia del más apto**
   En este paso, se eligen los individuos que pasarán a la siguiente generación. Los mejores individuos (aquellos con mayor fitness) tienen una mayor probabilidad de ser seleccionados, lo que refleja la idea de la "supervivencia del más apto". Existen diferentes métodos para la selección:
   - **Método de la ruleta**: Los individuos se seleccionan de manera probabilística, donde la probabilidad de ser seleccionado es proporcional a su valor de fitness.
   - **Torneo**: Un subconjunto aleatorio de individuos compite, y el ganador (con mejor fitness) es seleccionado.
   - **Selección por ranking**: Los individuos se ordenan según su fitness, y se seleccionan en función de su posición en el ranking.

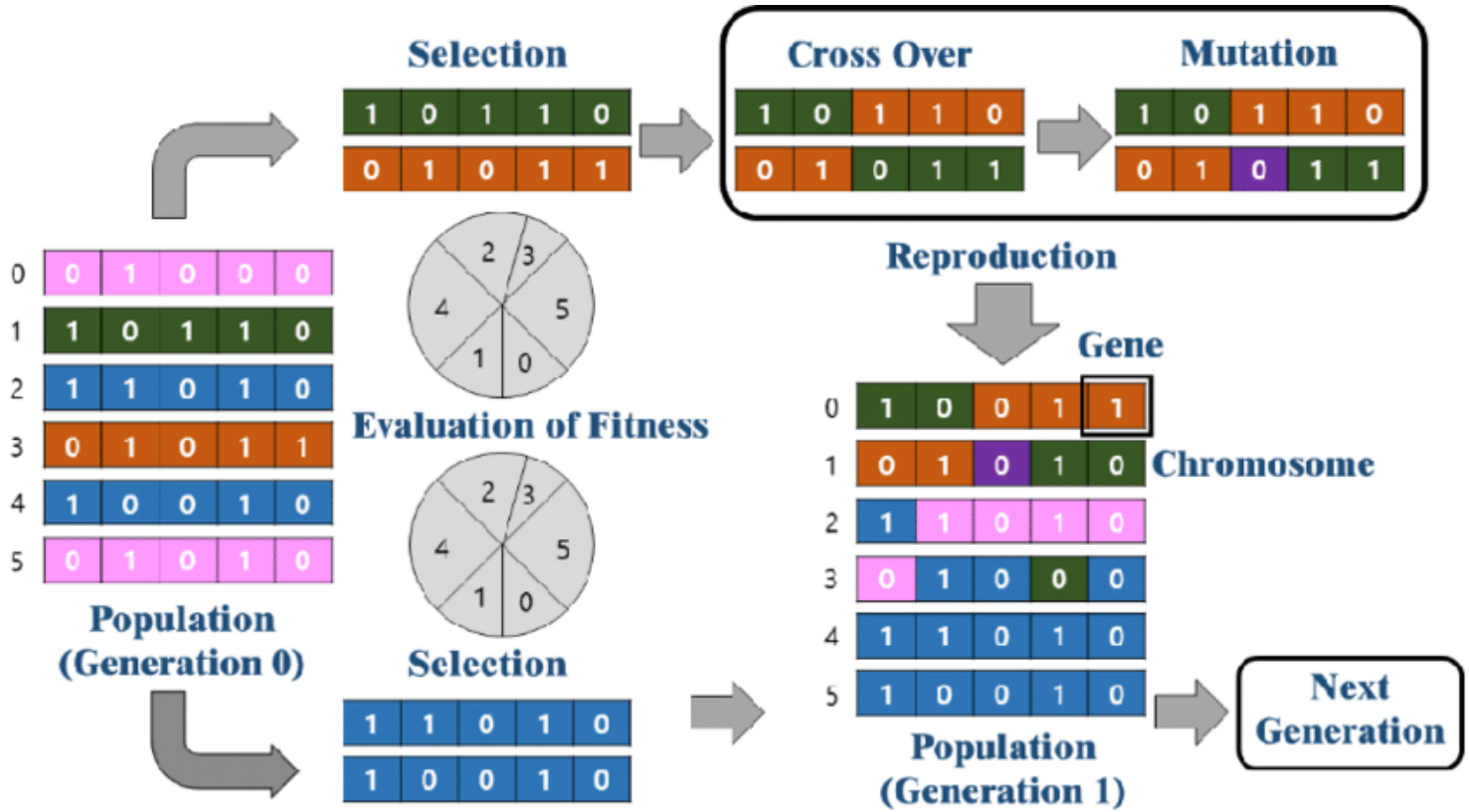
4. **Cruzamiento y mutación: Generación de nuevas soluciones**

   Cruzamiento: Después de seleccionar los individuos, se combina la información genética de dos padres para generar uno o más hijos.
   - **Cruzamiento de un punto**: Se selecciona un punto aleatorio en los cromosomas de los padres, y se intercambian las partes después de ese punto.
   - **Cruzamiento uniforme**: Cada gen se intercambia entre los padres con una probabilidad fija.

   Mutación: Tras el cruzamiento, algunos individuos sufren pequeñas modificaciones aleatorias. La mutación es clave para mantener la diversidad en la población y explorar nuevas áreas del espacio de soluciones.
   - **Mutación puntual**: Cambio de un valor o bit en un individuo.
   - **Mutación por desplazamiento**: Se cambia una parte del cromosoma por otra con un valor diferente.

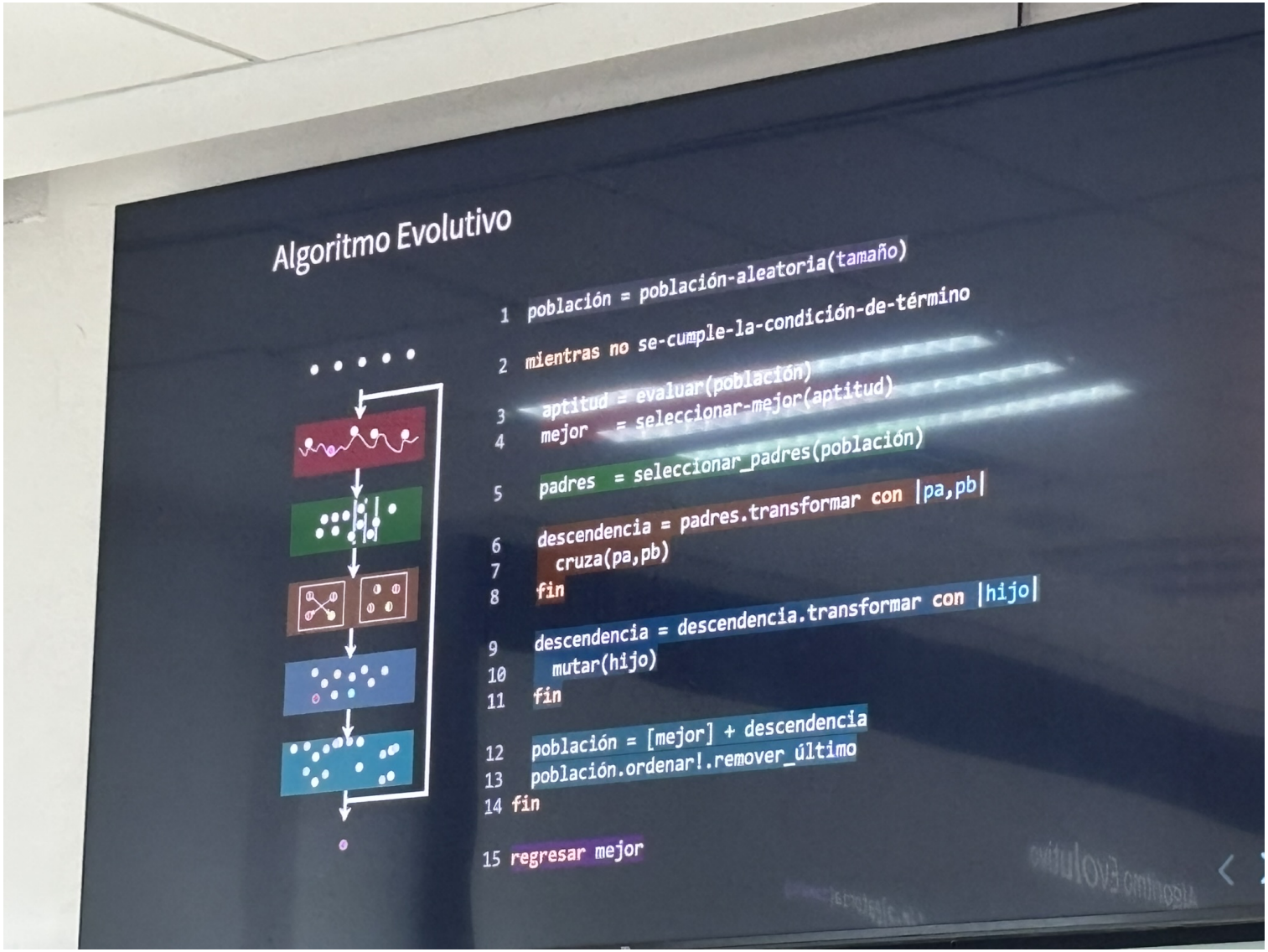
5. **Iteración: Repetición del ciclo hasta encontrar la solución óptima**
   El proceso de evaluación, selección, cruzamiento y mutación se repite durante varias generaciones. En cada iteración, la población evoluciona, mejorando sus soluciones hasta cumplir con uno de los siguientes criterios de parada:
   - **Número máximo de generaciones**: El algoritmo se detiene después de ejecutar un número fijo de iteraciones.
   - **Convergencia**: Si no se observan mejoras significativas en la población durante varias generaciones, se puede suponer que se ha alcanzado una solución cercana al óptimo.
   - **Criterio de fitness**: El algoritmo se detiene cuando se alcanza un valor de fitness predefinido que satisface las expectativas.


Start
Initialization
Evaluation
Crossover
Generation
Mutation
Selection
No
End?
Yes
Stop


Selection
Cross Over
Mutation
Reproduction
Gene
Chromosome
Evaluation of Fitness
Selection
Population
(Generation 0)
Population
(Generation 1)
Next
Generation

Fundamentos Teóricos de la Computación Evolutiva

Representación algorítmica



Ventajas y Desventajas de la Computación Evolutiva

- **Ventajas**:
  - Versatilidad para resolver problemas complejos.
  - Flexibilidad y capacidad de adaptación.
  - Funciona bien con problemas de optimización de múltiples objetivos.
- **Desventajas**:
  - Alto costo computacional.
  - Poca garantía de alcanzar la solución óptima (puede quedarse en óptimos locales).
  - Requiere ajuste manual de parámetros para un buen rendimiento.

Ejercicio Práctico: Algoritmo Evolutivo Simple

Descripción del problema

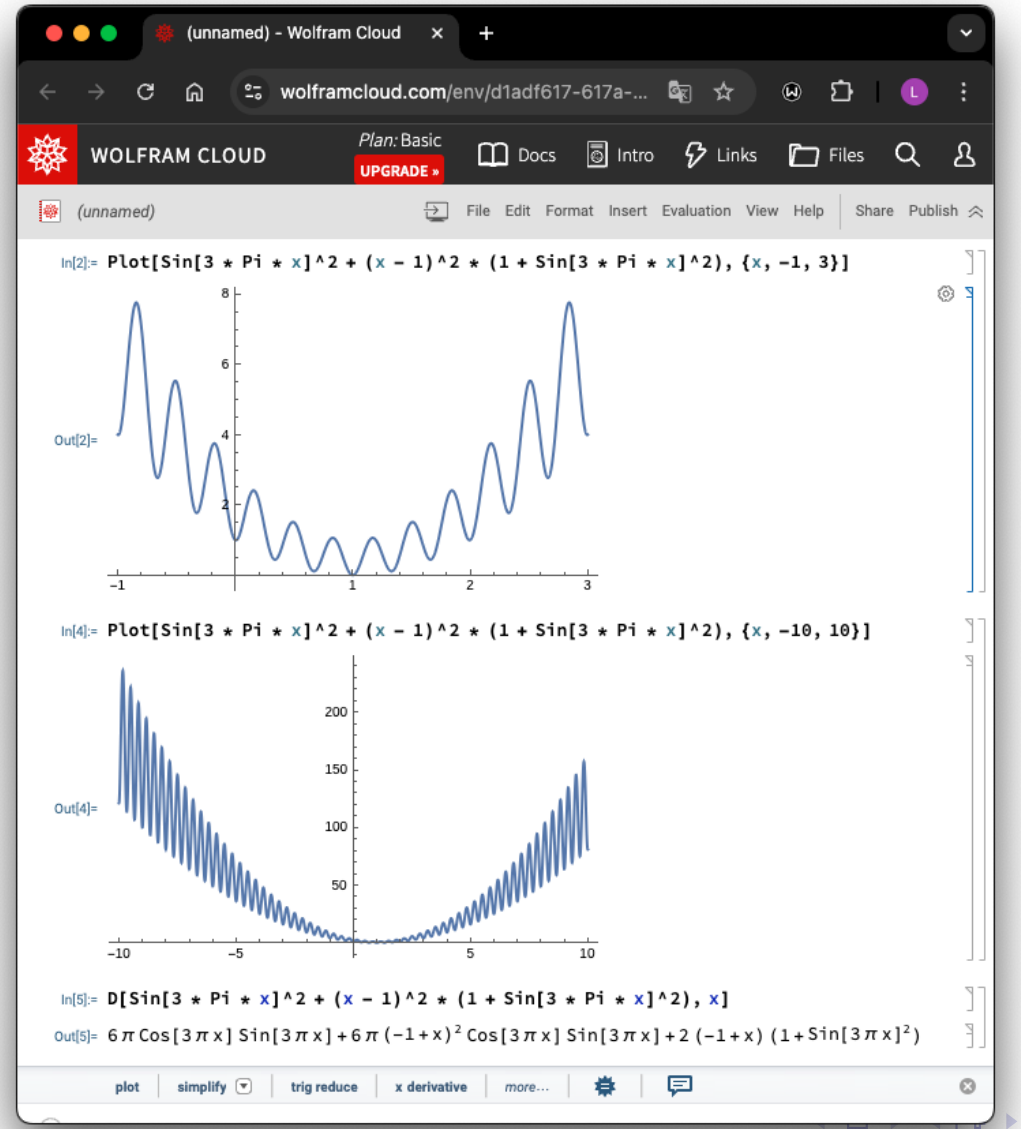
Enunciado

Optimizar la función:

$$f(x) = -(x-3)^2 + 10$$

Ejercicio Práctico: Algoritmo Evolutivo Simple

Análisis inicial



Referencias


Carlos Artemio Coello Coello (Editor) (2019)
Computación Evolutiva, Segunda Edición, Academia Mexicana de Computación, A.C.
ISBN: 978-607-97357-9-1
Disponible en: comp-evolutiva.pdf


Sakil Ansari (2020)
Hyperparameter tuning in Random Forest Classifier using genetic algorithm
Disponible en: Medium Article


Rodrigo Arenas (2021)
Tune Your Scikit-learn Model Using Evolutionary Algorithms
Publicado en: Towards Data Science
Disponible en: Towards Data Science


Udemy (2023)
Genetic Algorithm Courses on Udemy
Disponible en: Cursos en Udemy

Fin

¿Preguntas? ¿Comentarios?