Laboratorio 3 – Optimización y Búsqueda: Minimo global, TSP y Optimización de Horarios

Cristian Cative
Asignatura: Introducción a la Inteligencia Artificial
Universidad Santo Tomás

***Resumen*—Este informe describe la implementación y resultados del *Laboratorio 3*, que abarca tres problemas de optimización y búsqueda: (1) búsqueda del minimo global de una función mediante Hill Climbing; (2) resolución del Problema del Viajante (TSP) mediante un Algoritmo Genético sobre posiciones de ciudades; y (3) optimización de horarios escolares con un Algoritmo Genético que respeta restricciones de no solapamiento, disponibilidad de profesores y preferencias horarias. Se presenta la metodología, parámetros, resultados y conclusiones. El código fuente está incluido en el apéndice.**

***Index Terms*—Hill Climbing, Algoritmo Genético, TSP, Optimización de horarios, Inteligencia Artificial.**

I. Introducción

La optimización es central en muchas ramas de la ingeniería y la inteligencia artificial. En este laboratorio se abordan tres problemas representativos: encontrar minimos locales/globales de funciones (búsqueda local), resolver instancias del Problema del Viajante (TSP) con metaheurísticas y modelar la asignación de horarios escolares con restricciones reales por medio de algoritmos evolutivos. Estos ejercicios permiten familiarizarse con técnicas heurísticas, representación de soluciones y el manejo de restricciones reales en problemas combinatorios.

II. Objetivos

Objetivo general: Implementar y evaluar algoritmos de búsqueda y optimización para resolver tres problemas prácticos de IA.

Objetivos específicos:

- Implementar Hill Climbing para aproximar el máximo global de una función y visualizar la convergencia/punto máximo.
- Resolver una instancia de TSP con un Algoritmo Genético (selección, cruce OX, mutación por intercambio) y representar la mejor ruta trazada sobre la disposición de ciudades.
- Diseñar un Algoritmo Genético para optimizar horarios escolares que evite solapamientos, respete disponibilidades y preferencias, y evaluar su convergencia frente a distintas tasas de mutación.

III. Marco teórico

III-A. *Hill Climbing*

Hill Climbing es una búsqueda local que parte de una solución inicial y explora soluciones vecinas aceptando mejoras. Es simple y eficaz para funciones unimodales pero puede atascarse en óptimos locales.

III-B. *Algoritmos Genéticos (GA)*

Los GA son metaheurísticas inspiradas en la evolución. Operan sobre poblaciones de soluciones (individuos) mediante selección, cruce (crossover) y mutación. Son adecuados para problemas combinatorios (como TSP y asignación de recursos) por su capacidad de exploración/explotación.

III-C. *Problema del Viajante (TSP)*

TSP busca la ruta corta que visita cada ciudad exactamente una vez y regresa al origen. Es NP-hard y se beneficia de heurísticas y metaheurísticas.

IV. Metodología

Se implementaron los tres puntos en Python (VS Code). Bibliotecas: `numpy`, `matplotlib`, `random`, `copy` y opcionalmente `opencv-python` para detección de círculos (no necesaria en la versión final). A continuación se detalla cada punto.

IV-A. *Punto 1 – Mínimo global (Hill Climbing)*

Implementación: Se consideró la función objetivo

$$f(x) = x^2 - 3x + 4,$$

la cual es una parábola convexa con un *mínimo global* teórico en $x = 1{,}5$, $f(x) = 1{,}75$. El algoritmo de búsqueda local (*Hill Climbing*) se adaptó para este caso, de modo que en cada iteración se acepta un movimiento únicamente si el valor de la función disminuye.

Parámetros clave: se utilizó un número de iteraciones igual a 1000, con un paso aleatorio en el rango $[-0{,}1, 0{,}1]$. El punto inicial se selecciona de forma aleatoria dentro del intervalo $[-10, 10]$.

Visualización: Se graficó la función en el intervalo $[-10, 10]$ mostrando la curva de la parábola y resaltando el mínimo encontrado en color rojo junto con sus coordenadas.

IV-B. *Punto 2 – TSP con GA*

Representación: una ruta es una permutación de índices de ciudades. Las ciudades pueden definirse manualmente (coordenadas dadas en el código).

Operadores:

- *Fitness:* inverso de la distancia total (incluye regreso al origen).

- *Selección:* torneo entre dos individuos.
- *Cruce:* Ordered Crossover (OX).
- *Mutación:* intercambio (swap) entre dos posiciones con cierta probabilidad.

Ejecución: población de tamaño 60, 100 generaciones por defecto, se registra la mejor fitness por generación y se grafica la mejor ruta superpuesta a la disposición de ciudades con flechas dirigidas.

IV-C. Punto 3 – Optimización de horarios con GA

Representación: un individuo es un diccionario con claves por grupo (G1, G2, G3). Cada grupo contiene una lista de asignaciones por día (Lun..Vie): tuplas (materia, profesor).

Restricciones y penalizaciones:

- *Choques de profesor por slot:* penalización multiplicativa por asignación extra.
- *Profesor no disponible en slot:* penalización moderada.
- *Preferencias de materia:* penalización ligera si no se respeta.
- *Balance de carga:* penalización proporcional a la desviación estándar de horas por profesor.

Operadores GA: selección por torneo (size 5), cruce por grupos (cada grupo copia todo el bloque de uno de los padres), mutación que cambia materia o profesor en un slot.

Mejoras aplicadas: penalizaciones reducidas y parámetros aumentados (POP_SIZE=120, GENERATIONS=800) para mejorar la convergencia y obtener penalizaciones menores en la solución final.

V. Resultados

En esta sección se presentan los resultados obtenidos en el desarrollo de los tres puntos propuestos en el laboratorio. Cada uno de ellos corresponde a la aplicación de diferentes técnicas de Inteligencia Artificial y Optimización, implementadas en Python.

V-A. Punto 1 – Mínimo global (Hill Climbing)

Implementación: la función objetivo considerada fue:

$$f(x) = x^2 - 3x + 4,$$

la cual es una parábola convexa con un mínimo global teórico en

$$x = 1{,}5, \quad f(x) = 1{,}75.$$

Se aplicó el algoritmo de Búsqueda Local (*Hill Climbing*) con 1000 iteraciones, partiendo de un punto aleatorio en $[-10, 10]$ y realizando pequeños desplazamientos en cada paso. El algoritmo busca mover la solución hacia posiciones con menor valor de la función.

Parámetros clave: iteraciones = 1000; paso aleatorio en $[-0{,}1, 0{,}1]$.

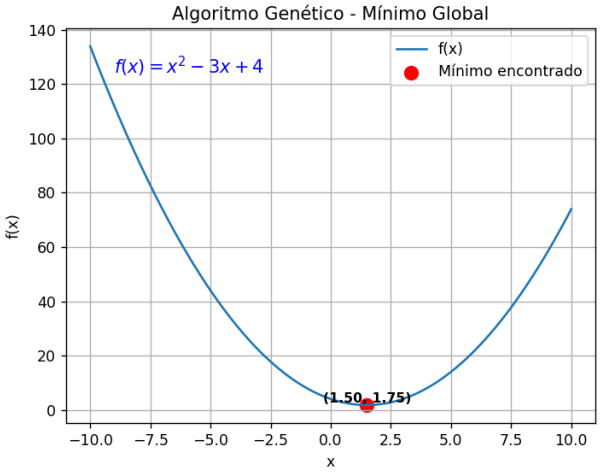

Figura 1: (Punto 1) Búsqueda Local de $f(x) = x^2 - 3x + 4$ mostrando el mínimo encontrado.

Al ejecutar el algoritmo, se obtuvo un mínimo aproximado en:

$$x = 1{,}50, \quad f(x) = 1{,}75,$$

lo cual coincide con el valor teórico esperado.

En la Figura 1 se observa que el punto en rojo corresponde al vértice de la parábola, confirmando que el algoritmo fue capaz de encontrar de manera precisa el mínimo global. Este resultado valida la eficacia del método en funciones unimodales convexas.

VI. Conclusiones

- Las heurísticas simples (Hill Climbing) resuelven problemas continuos sencillos con bajo costo computacional, pero son sensibles a óptimos locales.
- Los Algoritmos Genéticos demostraron ser versátiles para problemas combinatorios como TSP y asignación de horarios, siempre que la representación y los operadores se diseñen adecuadamente.
- En problemas con restricciones reales, el diseño de la función de penalización es crítico: pesos altos penalizan en exceso y dificultan la búsqueda; pesos moderados permiten que el GA mejore iterativamente.
- Ajustar población, generaciones y tasa de mutación es esencial: mayor población y generaciones ayudan a encontrar mejores soluciones, pero incrementan el coste computacional.

VI-A. Resultados del Punto 2 – Problema del Viajante (TSP)

Se realizaron dos configuraciones diferentes del conjunto de ciudades: la primera con 6 nodos y la segunda con 8 nodos. En ambos casos, se aplicaron los mismos parámetros de población, generaciones y tasa de mutación, lo cual permite observar la escalabilidad del algoritmo.

VI-A1. Caso con 6 ciudades: En la primera prueba, las ciudades fueron definidas en las coordenadas bidimensionales:

$$\{(0, 100), (40, 180), (300, 260), (100, 340), (100, 420), (200, 140)\}.$$

El algoritmo encontró como mejor solución la ruta:

$$[2, 5, 0, 1, 3, 4],$$

con una distancia total recorrida de aproximadamente $956{,}61$ unidades.

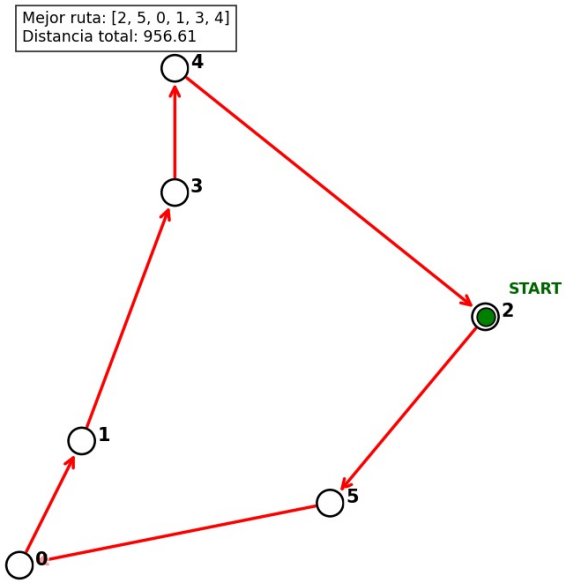

Figura 2: Mejor ruta obtenida para el TSP con 6 ciudades.

VI-A2. Caso con 8 ciudades: En la segunda prueba se amplió el número de nodos, considerando las coordenadas:

$$\{(30, 100), (40, 180), (600, 260), ..., (50, 340)\}.$$

El algoritmo encontró como mejor solución la ruta:

$$[6, 1, 0, 5, 2, 3, 4, 7],$$

con una distancia total recorrida de aproximadamente $1655{,}69$ unidades.

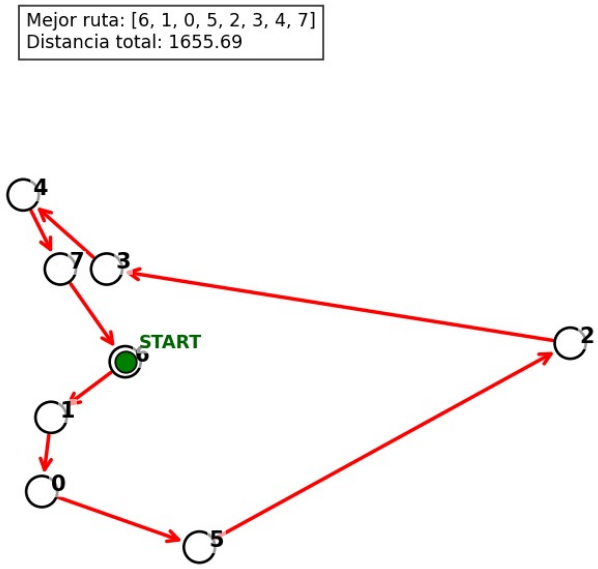

Figura 3: Mejor ruta obtenida para el TSP con 8 ciudades.

VI-A3. Análisis: El experimento evidencia que, aunque el algoritmo genético logra encontrar soluciones válidas para ambas configuraciones, la distancia total se incrementa de manera significativa con el aumento del número de nodos. Esto es consistente con la naturaleza del TSP, el cual es un problema NP-difícil, donde la complejidad crece factorialmente con el número de ciudades.

Asimismo, se resalta que la calidad de la solución depende directamente de los parámetros evolutivos (tamaño de población, número de generaciones y tasa de mutación). Aunque las soluciones no son necesariamente óptimas, representan una aproximación eficiente frente a métodos exactos, los cuales se vuelven inviables computacionalmente en problemas de mayor escala.

VI-B. Resultados del Punto 3 – Optimización de Horarios

En este caso, se aplicó el algoritmo genético para la construcción de un horario escolar sujeto a restricciones de disponibilidad de profesores, preferencias de materias y no solapamiento de asignaciones.

En la Figura 4 se muestra la comparación de la convergencia del algoritmo con diferentes tasas de mutación ($MR = 0{,}04$, $MR = 0{,}1$, $MR = 0{,}2$). Todas las configuraciones alcanzan un valor de fitness similar, aunque se observan diferencias en la rapidez de convergencia en las primeras generaciones.

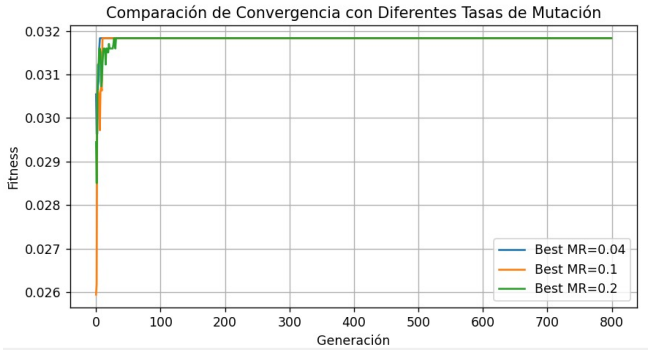

Figura 4: Comparación de la convergencia con distintas tasas de mutación.

Posteriormente, con la mejor configuración seleccionada, se obtuvo un valor de **fitness de 0.0318** y una **penalización de 30.41**. Estos resultados indican que, si bien el algoritmo logra construir horarios viables, persisten pequeños conflictos como asignaciones fuera de las preferencias ideales o ligeros desbalances en la carga docente.

El resultado final de la optimización se representa en la Figura 5, donde se observa la asignación completa de materias y profesores para los tres grupos (G1, G2 y G3) distribuidos de lunes a viernes.

Mejor Horario - Fitness=0.0318, Penalización=30.41

Grupo	Lun	Mar	Mié	Jue	Vie
G1	Inglés T_D	Física T_B	Arte T_D	Física T_B	Arte T_D
G2	Matemáticas T_A	Inglés T_D	Matemáticas T_A	Historia T_C	Física T_B
G3	Matemáticas T_A	Matemáticas T_A	Historia T_C	Inglés T_D	Historia T_C

Figura 5: Horario óptimo generado con el Algoritmo Genético.

VI-B1. Análisis: Los resultados muestran que el algoritmo genético puede producir horarios factibles en escenarios con múltiples restricciones. La gráfica de convergencia evidencia que valores de mutación bajos ($MR = 0{,}04$) generan una búsqueda más estable, mientras que tasas más altas ($MR = 0{,}2$) incrementan la exploración pero también introducen más variabilidad al inicio.

El valor de penalización obtenido (30.41) refleja que la solución final no es perfecta, pero representa un compromiso aceptable entre la satisfacción de restricciones y la viabilidad computacional. El horario generado constituye una solución práctica, aunque mejorable mediante ajustes en los parámetros evolutivos o la ponderación de penalizaciones.

VII. Conclusiones

- El uso de Hill Climbing permitió identificar de manera eficiente el mínimo global de una función cuadrática. Se evidenció que este tipo de algoritmos son apropiados en problemas unimodales, pero presentan limitaciones frente a funciones multimodales debido a la posibilidad de quedar atrapados en óptimos locales.
- El Algoritmo Genético aplicado al TSP mostró ser una herramienta flexible para encontrar rutas aproximadas de buena calidad en tiempos razonables. Aunque las soluciones obtenidas no garantizan optimalidad, su desempeño frente al crecimiento exponencial de la complejidad del problema confirma la utilidad de las metaheurísticas en problemas NP-difíciles.
- La construcción de horarios escolares mediante un Algoritmo Genético demostró la capacidad de este enfoque para manejar múltiples restricciones simultáneamente (disponibilidad, preferencias y balance de carga). Si bien el resultado presentó una penalización moderada, se logró un horario factible, lo cual valida la eficacia de los GA en problemas combinatorios de la vida real.

Referencias

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, 1992.
[2] T. Bektas, "The multiple traveling salesman problem: A survey," *European Journal of Operational Research*, vol. 200, no. 1, pp. 1–10, 2006.
[3] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, 2nd ed., Springer, 2015.