Informe Técnico: Algoritmos Genéticos aplicados a CartPole-v1

Cristopher Acuña Campos — 2022437718

Jerson Prendas Quirós — 2022437678

Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

Inteligencia Artificial

PROF. Kenneth Obando Rodriguez

23/09/2025

Resumen Ejecutivo

Este informe documenta el desarrollo e implementación de un algoritmo genético (AG) aplicado al entorno **CartPole-v1** de Gymnasium. El objetivo fue optimizar el comportamiento de un agente que mantiene en equilibrio un palo sobre un carro. Se exploraron tres configuraciones experimentales variando población y tasa de mutación, con el fin de comparar la convergencia y estabilidad del aprendizaje. Los resultados demuestran que el AG logra entrenar agentes capaces de mantener el equilibrio durante el episodio completo (500 pasos), validando la efectividad del enfoque evolutivo.

Introducción

Los algoritmos genéticos son metaheurísticas inspiradas en la selección natural. Simulan procesos evolutivos como **selección**, **cruce** y **mutación** para optimizar soluciones en espacios complejos. Son especialmente útiles cuando no se dispone de gradientes o modelos exactos del problema.

El entorno **CartPole-v1** plantea un reto de control clásico: mantener un palo en equilibrio sobre un carro aplicando fuerzas discretas a izquierda o derecha. Aunque es un problema relativamente simple para técnicas como Q-Learning, usar algoritmos genéticos permite ilustrar el poder de la optimización estocástica en tareas de control.

Este trabajo busca reforzar competencias en representación genética, diseño de funciones de fitness y análisis experimental.

Descripción de la modalidad escogida

Se eligió la modalidad de juego, aplicando el AG al entorno **CartPole-v1** de Gymnasium. Este escenario permite evaluar de forma objetiva la calidad de un agente midiendo su recompensa acumulada. Además, facilita la visualización de resultados, requisito esencial para comprender la evolución de la población.

Diseño del cromosoma y función de fitness

- **Cromosoma:** Vector de 4 de pesos reales que pondera las 4 variables de estado de Cart Pole (posición del carro, velocidad, ángulo del palo y velocidad angular).
- **Política:** El agente calcula la acción evaluando el signo del producto punto entre los pesos y el vector de observaciones. Si el resultado es positivo, aplica fuerza hacia la derecha; en caso contrario, hacia la izquierda.

- **Fitness:** Recompensa total acumulada durante el episodio de simulación. El episodio termina si el palo cae, si se sale de la pantalla de simulación o si se alcanzan los 500 pasos, o sea, se supera la prueba.

Parámetros y configuración experimental

Se implementó un AG con:

- **Población**: Se refiere al número de soluciones candidatas por generación. Entre más grande sea, mejora la exploración del espacio de búsqueda y reduce la varianza, pero aumenta el coste de cómputo por generación. Se encuentra definida por la constante AMOUNT_OF_AGENTS, inicialmente está predefinido en **30** pero puede variar. En la fase de experimentacion se le asignan valores de 30 y 50.
- **Generaciones**: Cuántas "rondas" evolutivas se ejecutan. Más generaciones permiten refinar y consolidar buenas soluciones. Nuestro entorno la define como AMOUNT_OF_GENERATIONS y se encuentra configurado inicialmente con un valor de 50, aunque se realizaron pruebas con valores 100 de igual manera.
- **Mutación**: Probabilidad de alterar cada "gen" (cada peso) al generar descendencia. Controla la exploración: valores bajos (≈0.05–0.2) preservan lo aprendido; valores altos (≈0.3–0.5) exploran más pero pueden desestabilizar. Es representado por la constante MUTATION_RATE = 0.2 (variantes: 0.5).
- **Selección**: Conocido como "elitismo", se refiere al porcentaje de individuos con mayor rendimiento que se preservan directamente en la siguiente generación. Un 10% suele equilibrar explotación y diversidad. Muy alto reduce la diversidad; muy bajo diluye lo aprendido.
- **Cruce**: Mecanismo para **mezclar** características de dos buenos individuos creando hijos. Al repartir "genes" de ambos padres, se combinan rasgos útiles ya descubiertos. En nuestro caso cada peso del hijo se toma, con probabilidad 50%, el alelo del padre 1 o del padre 2.
- **Reemplazo generacional**: nueva generación formada por
    - 10% élite sin alterar,
    - 30% cruce de élite mezclada,
    - 30% copias mutadas de la élite,
    - 30% mutación aplicada a hijos de cruce.

Métricas

Se evaluaron cuatro configuraciones de prueba:

Configuración	Población	Generaciones	Mutación
Población Insuficiente	20	30	0.2
Base	30	50	0.2
Mutación Alta	30	50	0.5
Mayor Población	50	100	0.2

Por generación, se computan max_fitness_in_gen[g] = max(fitness_values) y avg_fitness_in_gen[g] = mean(fitness_values). En nuestro caso **analisis_estadistico** imprime los valores correspondientes a cada generación y crea una gráfica utilizando matplotlib, esto basado en los resultados obtenidos tras simular el entorno con las constantes configuradas en **cart_pole_v1.**

Resultados y visualizaciones

Durante la experimentación se registraron, por cada generación, tanto el fitness máximo alcanzado como el fitness promedio de la población. Estos indicadores permiten observar tanto el desempeño de los mejores individuos como la estabilidad general de la población a lo largo del proceso evolutivo.

Población Insuficiente

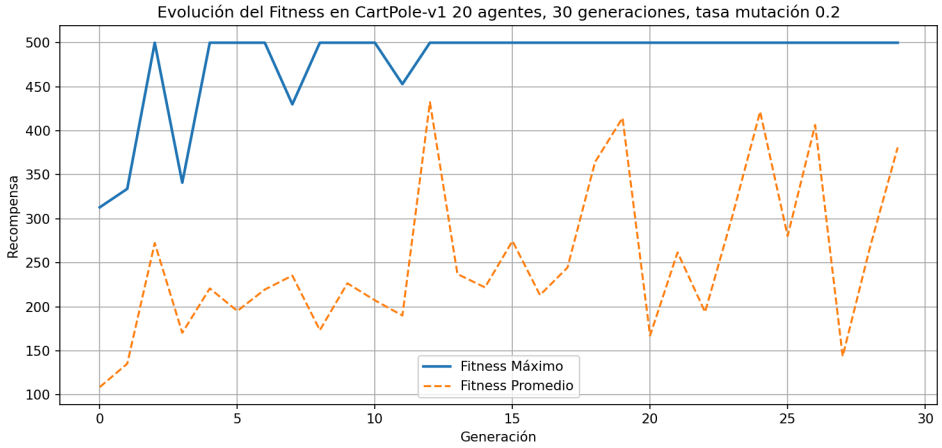

Figura 1. Evolución del fitness en la configuración Población Insuficiente. Fuente: elaboración propia con Matplotlib.

La gráfica muestra que algunos individuos alcanzaron rápidamente el máximo de 500 pasos, pero el fitness promedio permaneció bajo y con gran dispersión. Esto indica que, aunque surgieron agentes exitosos, estos fueron minoría frente al gran número de población que no logró mejorar de manera consistente debido al tamaño reducido y al bajo número de generaciones. Son evidentes varios retrocesos en cuanto a puntuaciones máximas, mostrados con esos picos bajos antes y después de la generación 10.

Configuración Base

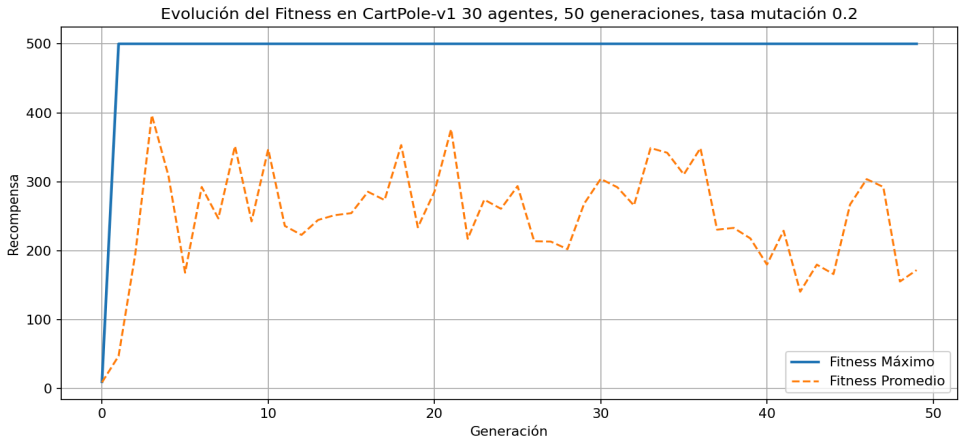

Figura 2. Evolución del fitness en la configuración Base. Fuente: elaboración propia con Matplotlib.

El fitness máximo alcanzó los 500 pasos de forma temprana, antes de las primeras 5 generaciones y se mantuvo estable. El promedio poblacional mostró una tendencia más o menos equilibrada, con menos oscilaciones que en el caso anterior aunque sí parece que sobre la generación 40 tomó una tendencia descendente. Aún así, esto sugiere un mejor balance entre diversidad y aprovechamiento de los mejores individuos.

Mutación Alta

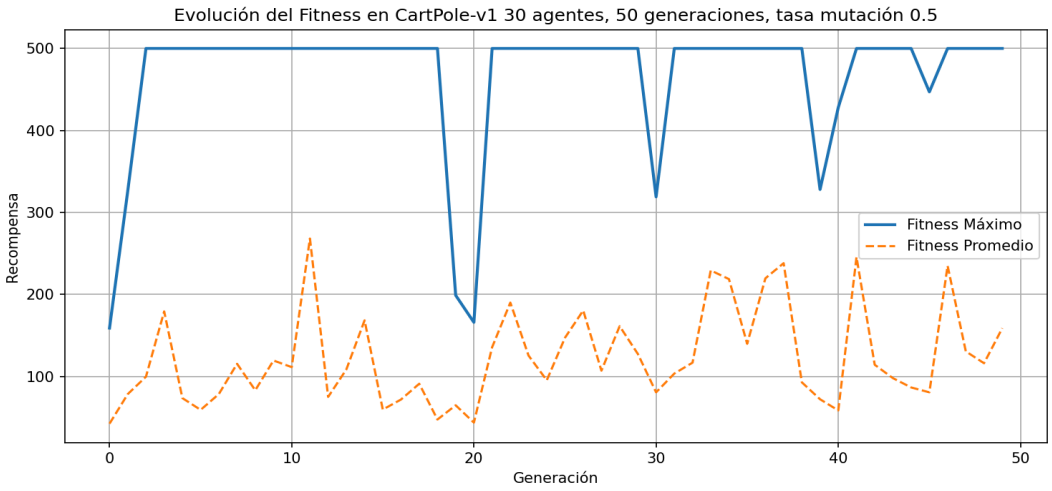

Figura 3. Evolución del fitness en la configuración Mutación Alta. Fuente: elaboración propia con Matplotlib.

La curva del fitness máximo evidenció fluctuaciones notorias, alcanzando en algunas generaciones el valor óptimo y descendiendo en otras. El promedio también resultó inestable, con variaciones frecuentes. Estos resultados reflejan que una tasa de mutación demasiado elevada introduce ruido excesivo en la población, dificultando la consolidación de soluciones sólidas.

Mayor Población

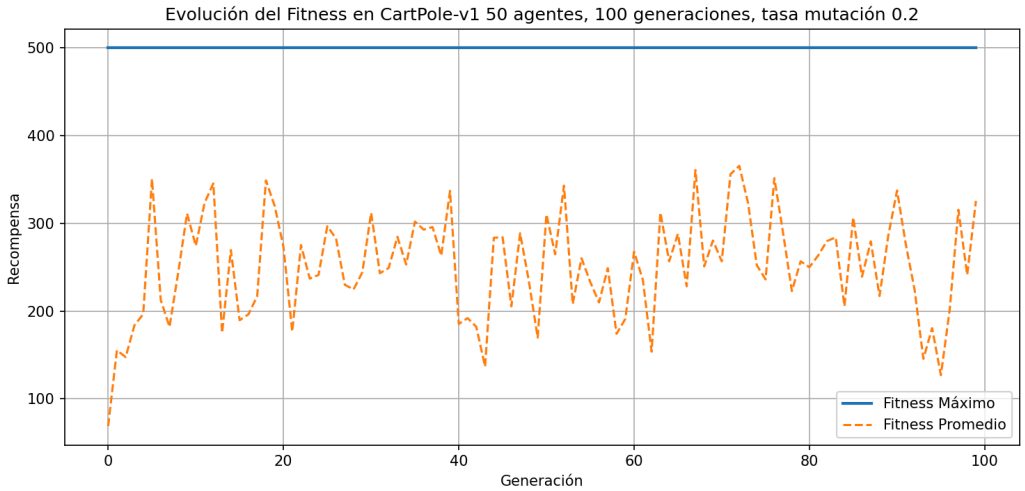

Figura 4. Evolución del fitness en la configuración Población Alta. Fuente: elaboración propia con Matplotlib.

Este escenario presentó el comportamiento más estable. El fitness máximo alcanzó los 500 pasos desde las primeras generaciones y se mantuvo constante. El fitness promedio, aunque

sin llegar a valores cercanos al máximo, mostró una tendencia más uniforme y con menor dispersión que en los demás experimentos. Esto confirma que poblaciones más grandes y un mayor número de generaciones favorecen la estabilidad y robustez del proceso evolutivo.

En conclusión, las gráficas evidencian que el algoritmo genético es capaz de entrenar agentes que resuelven CartPole-v1, pero la calidad y consistencia de los resultados dependen fuertemente de los parámetros de población y mutación.

Discusión crítica y análisis

Los resultados obtenidos permiten reflexionar sobre el comportamiento del algoritmo genético bajo diferentes configuraciones y las implicaciones de los parámetros de diseño. Un primer aspecto clave es la influencia del tamaño de la población. En la configuración de población insuficiente se observó que, aunque surgieron individuos capaces de resolver el entorno de forma aislada, el promedio poblacional se mantuvo bajo. Esto evidencia que poblaciones pequeñas carecen de la diversidad genética necesaria para sostener un progreso colectivo. Los individuos exitosos no logran arrastrar al resto de la población, lo cual se traduce en fluctuaciones marcadas y una convergencia deficiente.

En contraste, al aumentar la población a 30 agentes (configuración base) se logró un equilibrio más adecuado. El promedio de la población creció con mayor estabilidad, lo cual sugiere que la mayor diversidad permitió transmitir características beneficiosas a un número mayor de descendientes. El efecto se intensificó aún más en la configuración de 50 agentes, donde la diversidad adicional contribuyó a reducir el ruido estocástico y favorecer una convergencia más uniforme. Estos hallazgos confirman la importancia de mantener poblaciones suficientemente grandes en algoritmos genéticos, pues un número reducido de individuos puede llevar a resultados engañosos en los que se logra un buen desempeño puntual, pero no se consolida un aprendizaje generalizado.

El segundo parámetro relevante es la tasa de mutación. En la configuración con valor 0.2 se alcanzó un equilibrio entre exploración y explotación. Este nivel de mutación permitió introducir suficiente variabilidad como para evitar un estancamiento prematuro, pero sin deteriorar de manera abrupta las soluciones encontradas. Por el contrario, al incrementar la mutación a 0.5, las gráficas mostraron un patrón mucho más errático. Aunque en ciertas generaciones se alcanzó el rendimiento óptimo de 500 pasos, en otras se produjeron caídas bruscas, indicando que los descendientes no heredan ni conservan adecuadamente los rasgos de los mejores individuos. Esto evidencia cómo tasas de mutación demasiado altas pueden tener un efecto desestabilizador, comprometiendo la consolidación de soluciones sólidas.

Otro aspecto relevante es el número de generaciones. Con 100 generaciones, en la configuración de mayor población, no solo se alcanzó de forma rápida el máximo desempeño,

sino que se mantuvo de manera estable a lo largo de toda la simulación. Esto muestra que extender el número de iteraciones del proceso evolutivo otorga más oportunidades para que las soluciones se refinen y se transmitan con mayor consistencia a las siguientes generaciones.

En conjunto, los resultados confirman que el algoritmo genético es una herramienta eficaz para resolver un entorno clásico de control como CartPole-v1 incluso con políticas simples basadas en pesos lineales. No obstante, también evidencian que el éxito depende fuertemente de la calibración de parámetros. Poblaciones grandes, tasas de mutación moderadas y un número suficiente de generaciones ofrecen un balance adecuado entre diversidad y explotación, mientras que configuraciones extremas (poblaciones pequeñas o mutaciones muy altas) conducen a inestabilidad o a una convergencia limitada.

Finalmente, cabe señalar una limitación del trabajo: la política empleada es lineal, lo que restringe la complejidad de los comportamientos aprendidos. Aunque fue suficiente para CartPole-v1, entornos más desafiantes probablemente requieran cromosomas más expresivos, como pequeñas redes neuronales. Asimismo, se podrían explorar variantes del algoritmo como tasa de mutación adaptativa, que podrían mejorar la preservación de los mejores individuos y optimizar el uso de recursos computacionales.

Conclusiones y trabajo futuro

Los experimentos realizados demuestran que los algoritmos genéticos son una estrategia viable para aprender políticas de control en entornos clásicos como CartPole-v1. A pesar de utilizar una representación cromosómica simple basada en pesos lineales, el sistema logró entrenar agentes capaces de mantener el equilibrio durante los 500 pasos máximos permitidos, cumpliendo así el objetivo de la simulación correctamente.

Entre los principales hallazgos se destaca que:

- **El tamaño de la población** influye directamente en la estabilidad del aprendizaje. Poblaciones pequeñas permiten la aparición de individuos óptimos aislados, pero no garantizan mejoras generalizadas. En cambio, poblaciones más grandes aportan diversidad y reducen el riesgo de convergencia prematura.
- **La tasa de mutación** actúa como regulador entre exploración y explotación. Con valores moderados (≈0.2) se logró un equilibrio que permitió progresos estables. Con valores altos (≈0.5) se introdujo demasiada variabilidad, lo que dificultó la consolidación de soluciones.
- **El número de generaciones** tiene un efecto claro en la consolidación del aprendizaje. Simulaciones prolongadas (100 generaciones) favorecieron la estabilidad y la propagación de soluciones efectivas, aunque a un coste computacional mayor.

En términos generales, los resultados confirman que la calibración de parámetros es fundamental para garantizar un balance adecuado entre diversidad genética, velocidad de convergencia y eficiencia computacional.

Trabajo futuro

A partir de lo observado, se identifican varias oportunidades de mejora:

1. **Políticas más expresivas**: sustituir los cromosomas lineales por pequeñas redes neuronales, lo cual permitiría resolver entornos más complejos donde las relaciones entre variables no son lineales.
2. **Mutación adaptativa**: variar la tasa de mutación en función del progreso evolutivo, manteniendo diversidad en las etapas iniciales y reduciendo el ruido en fases de explotación.
3. **Extensión a otros entornos**: aplicar el enfoque a otros problemas de igual o mayor dificultad, como MountainCar-v0 o Acrobot-v1, donde la dinámica exige políticas más complejas.

En síntesis, el trabajo confirma la utilidad de los algoritmos genéticos como herramienta de optimización estocástica en problemas de control, al tiempo que abre la puerta a mejoras metodológicas y técnicas que ampliarían su aplicabilidad y eficiencia en escenarios más demandantes.