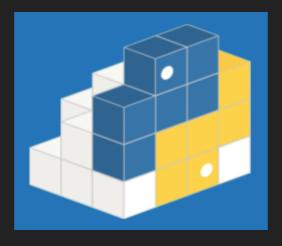
BIBLIOTECAS PARA ALGORITMOS GENÉTICOS

DEAP 1.4.3

- ▶ DEAP (Distributed Evolutionary Algorithms in Python).
- Algoritmo genético que utiliza representación lista, matriz, conjunto, diccionario, árbol, matriz
 Numpy, etc.
- > Programación genética utilizando árboles de prefijos tipo flexible y tipo fuerte.
- ▶ Estrategias de evolutivas (CMA-ES).
- Description de enjambre de partículas y Evolución diferencial.
- Optimización multiobjetivo (NSGA-II, NSGA-III, SPEA2, MO-CMA-ES).
- ▶ Funciona con mecanismos de paralelización como el multiprocesamiento.



DEAP 1.4.3



from deap import base, creator, tools, algorithms

```
creator.create("FitnessMin", base.Fitness, weights=(-1.0,))
creator.create("Individual", list, fitness=creator.FitnessMin)
toolbox = base.Toolbox()
# Atributo generador (gen): 0 o 1
toolbox.register("attr bool", random.randint, 0, 1)
TOTAL N BITS = N BITS PER VARIABLE * 2
toolbox.register("individual", tools.initRepeat, creator.Individual, toolbox.attr bool, n=TOTAL N BITS)
toolbox.register("population", tools.initRepeat, list, toolbox.individual)
toolbox.register("evaluate", evalParaboloidBinary)
toolbox.register("mate", tools.cxTwoPoint) # Cruce de dos puntos
toolbox.register("mutate", tools.mutFlipBit, indpb=0.05) # Probabilidad de mutar cada bit
toolbox.register("select", tools.selTournament, tournsize=3)
```

DEAP 1.4.3



```
# --- 3. Ejecución del Algoritmo Genético ---
pop_size = 100
cruceprob = 0.7
mutacionprob = 0.2
num_generaciones = 100

pop = toolbox.population(n=pop_size)
hof = tools.HallOfFame(1)

pop, logbook = algorithms.eaSimple(pop, toolbox, cxpb=cruceprob, mutpb=mutacionprob, ngen=num_generaciones, halloffame=hof, verbose=True)
```

 PyGAD es un Biblioteca de Python de código abierto para construir el algoritmo genético.

Problemas de optimización mono-objetivo y multiobjetivo.



```
!pip install pygad
import pygad
```

```
# --- 2. Configuración del Algoritmo Genético con PyGAD (más parecido a DEAP) ---
# Parámetros del AG
num_generations = 100
num_parents_mating = 10
sol_per_pop = 50
num_genes = 2
# Límites para cada gen (x e y)
gene_space = [{'low': BOUND_LOW, 'high': BOUND_UP}, {'low': BOUND_LOW, 'high': BOUND_UP}]
# Parámetro K para la selección por torneo (tamaño del torneo)
K_tournament = 3
```





Cruza multipunto flotante:

Padre 1 = [1.0, 2.5, 3.1, 4.8, 5.2]

Padre 2 = [0.5, 1.1, 6.3, 0.9, 2.0]

Padre 1: [1.0 | 2.5, 3.1 | 4.8, 5.2]

Padre 2: [0.5 | 1.1, 6.3 | 0.9, 2.0]

Hijo 1 = [1.0, 1.1, 6.3, 4.8, 5.2]

Hijo 2 = [0.5, 2.5, 3.1, 0.9, 2.0]





REFERENCIAS BIBLIOGRÁFICAS Y WEB (I)

- https://pypi.org/project/deap/
- https://pygad.readthedocs.io/en/latest/

