Explicación del Código y Proceso NSGA-II

El algoritmo NSGA-II (Non-dominated Sorting Genetic Algorithm II) es un algoritmo evolutivo de optimización multiobjetivo que se utiliza para encontrar un conjunto de soluciones óptimas, conocidas como *frente de Pareto*, donde no existe una solución que domine a otra. En otras palabras, se busca encontrar soluciones donde no se pueda mejorar un objetivo sin empeorar otro. El proceso de este algoritmo incluye varias fases clave, y con base en el código que proporcionas, aquí te explico cada uno de los pasos que se están ejecutando:

Explicación del Código y Proceso NSGA-II

1. Inicialización de la Población:

En el paso 1 se crea una población aleatoria de soluciones (en este caso, puntos bidimensionales representados por las coordenadas `x1` y `x2`, que están en el dominio de `[0, π]`).

```
population = [ [random.uniform(0, math.pi), random.uniform(0, math.pi)] for
_ in range(population_size) ]
```

Se generan soluciones con valores aleatorios dentro del rango definido por `0 <= x1, x2 <= π`. La población inicial contiene `population_size` individuos.

Ejemplo de población inicial (STEP 1):

```
POPULATION = [[2.205433294806139, 1.524425477512366], [1.4246693878960441,
1.6021238868432237], ...]
```

2. Evaluación de la Población:

En este paso (STEP 2), cada solución es evaluada utilizando las funciones objetivo definidas en el código (`min_f1(x1, x2)` y `min_f2(x1, x2)`), pero antes se comprueban las restricciones (funciones `C1(x1, x2)` y `C2(x1, x2)`). Si la solución no cumple con las restricciones, se le asignan valores de fitness infinitos, lo que la hace inviable.

```
evaluated_population = [ ind + evaluate_solution(ind[0], ind[1]) for ind in
population ]
```

Cada individuo obtiene un vector de 4 valores: sus dos coordenadas `(x1, x2)` y sus valores de objetivo `(f1, f2)`.

Ejemplo de población evaluada (STEP 2):

```
EVALUATED POPULATION = [[2.205433294806139, 1.524425477512366, inf, inf],
...]
```

3. Clasificación en Frentes de Pareto:

El objetivo es clasificar la población en varios frentes de Pareto. En este paso, las soluciones que no son dominadas por ninguna otra se agrupan en el primer frente de Pareto. Luego, las soluciones dominadas por el primer frente se agrupan en el siguiente frente, y así sucesivamente. El frente de Pareto contiene las soluciones no dominadas, es decir, aquellas que no pueden ser mejoradas en ningún objetivo sin empeorar otro.

```
pareto_front = find_pareto_frontier(evaluated_population)
```

Ejemplo de un frente de Pareto (STEP 3):

```
PARETO FRONT = [[2.205433294806139, 1.524425477512366, inf, inf], ...]
```

4. Selección de la Nueva Población:

En este paso, las soluciones que forman el frente de Pareto se seleccionan para formar la nueva población. Los individuos seleccionados no incluyen los valores de los objetivos, solo las coordenadas `(x1, x2)` se mantienen.

```
population = [ind[:2] for ind in pareto_front]
```

Ejemplo de nueva población (STEP 4):

```
NEW POPULATION = [[2.205433294806139, 1.524425477512366], ...]
```

5. Generación de Descendencia (Crossover y Mutación):

El siguiente paso consiste en generar nueva descendencia a partir de la población seleccionada. Se utiliza el operador de cruce (crossover) para combinar dos soluciones y generar una nueva solución, y luego se aplica una pequeña mutación para introducir variabilidad. Este proceso se repite hasta que se ha generado suficiente descendencia para llenar la población.

```
offspring = [] while len(offspring) < population_size: p1, p2 = random.samp
le(population, 2) child = mutate(crossover(p1, p2)) offspring.append(child)
```

Ejemplo de descendencia generada (STEP 5):

```
OFFSPRING = [[1.6231683139580477, 2.1978903743321885], ...]
```

6. Reemplazo de la Población:

Finalmente, la población se actualiza combinando la población original con la descendencia generada. Para mantener el tamaño de la población constante, se recorta al número de individuos especificado (`population_size`).

Resumen de las Etapas en Cada Iteración

- **Paso 1 (Población)**: Se crea la población inicial.
- **Paso 2 (Evaluación)**: Se evalúan las soluciones con respecto a los objetivos y las restricciones.
- **Paso 3 (Frente de Pareto)**: Se identifica el frente de Pareto con las soluciones no dominadas.
- **Paso 4 (Nueva Población)**: Se seleccionan las soluciones no dominadas para la siguiente generación.
- **Paso 5 (Descendencia)**: Se generan nuevos individuos mediante cruce y mutación.
- **Paso 6 (Reemplazo)**: La población se actualiza con los individuos seleccionados y la descendencia.

Resumen del proceso:

1. **Inicialización**: Se genera una población inicial de soluciones aleatorias.
2. **Evaluación**: Las soluciones son evaluadas en función de sus objetivos y restricciones.
3. **Clasificación**: Se organizan las soluciones en frentes de Pareto, buscando las no dominadas.
4. **Selección**: Se seleccionan las soluciones utilizando la distancia euclidiana al centroide del frente de Pareto.
5. **Cruce y Mutación**: Se generan nuevos individuos aplicando cruce y mutación a los seleccionados.
6. **Reemplazo**: La población se actualiza con las soluciones actuales y la descendencia generada.
7. **Visualización**: Se visualiza la evolución de la frontera de Pareto a través de las generaciones.

Este proceso se repite a lo largo de varias generaciones, evolucionando hacia un conjunto de soluciones que representan la **frontera de Pareto** en un problema de optimización multiobjetivo.