EC Assignment #4 (NSGA-II)

電機系（碩一）陳昕佑 61375017H

The **NSGA-II** (Non-dominated Sorting Genetic Algorithm II) is a popular multi-objective optimization algorithm designed to solve problems like the ZDT suite effectively, it follows these main steps:

1. **Initialization:**
   Population: Generate an initial population of size N with random solutions within the problem's bounds.
   Representation: Each individual represents a solution with decision variables $x = (x_1, \dots, x_n)$.
   Evaluation: Compute the objective functions $f_1(x), \dots, f_k(x)$ for all individuals.
2. **Non-dominated Sorting:**
   Definition: A solution a dominates b if:
   $f_i(a) \le f_i(b) \forall i$ (better or equal in all objectives).
   $f_i(a) < f_i(b)$ for at least one i (strictly better in at least one objective).
   Procedure: Sort the population into **Pareto fronts**
   Front 1 (F_1): Non-dominated solutions.
   Front 2 (F_2): Solutions dominated by F_1 but not by each other.
   Continue for all fronts.
3. **Crowding Distance:**
   Purpose: Ensures diversity by maintaining a spread of solutions on the Pareto front. For each front:
   For each objective i, sort solutions by f_i.
   Compute crowding distance for solution j: $d_j = \sum_{i=1}^{k} \frac{f_i^{j+1} - f_i^{j-1}}{f_i^{max} - f_i^{min}}$
   where f_i^{max} and f_i^{min} are the max/min values of f_i.
   Boundary solutions are assigned infinite distance to preserve edge diversity.

4. **Selection (Binary Tournament):**
   Select N individuals for mating based on:
   Rank: Solutions in better (lower-rank) fronts are preferred.
   Crowding Distance: Among solutions of the same rank, those with higher crowding distances are preferred.
5. **Crossover and Mutation:**
   Generate offspring by combining parent solutions:
   Simulated Binary Crossover (SBX): Combines two parents to create new offspring by blending their variables.
   Polynomial Mutation: Perturbs a single solution by adding small random changes to its variables.
   Offspring solutions inherit properties from parents, allowing exploration and exploitation of the search space.
6. **Environmental Selection:**
   Combine parents and offspring into a combined population $(2N)$.
   Perform non-dominated sorting on the combined population.
   Select the top N solutions based on:
   Rank: Lower-ranked solutions are preferred.
   Crowding Distance: Preserves diversity.
7. **Termination:**
   Repeat steps 3–6 until the stopping criterion is met (e.g., maximum evaluations or generations).
   Output the final population, which approximates the Pareto front.

How NSGA-II Solves ZDT Problems

1. **Handling Multiple Objectives**
   NSGA-II maintains a **Pareto-optimal set** of solutions by performing non-dominated sorting.
   For ZDT problems, it simultaneously optimizes $f_1(x)$ and $f_2(x)$ while balancing:
   Convergence: Ensuring solutions reach the Pareto front.
   Diversity: Spreading solutions across the Pareto front.
2. **Challenges Addressed**
   ZDT1 (Convex Front): NSGA-II's diversity preservation ensures a uniform distribution across the simple convex front.
   ZDT2 (Non-convex Front): The algorithm's rank and crowding distance mechanisms work together to converge to the complex front.
   ZDT3 (Disconnected Front): NSGA-II's ability to preserve solutions in different Pareto fronts ensures it finds and maintains the disjoint segments.
   ZDT4 (Multimodal Landscape): Exploration mechanisms (mutation and crossover) allow NSGA-II to escape local optima and converge to the Pareto front.
   ZDT6 (Non-uniform Distribution): NSGA-II adapts to the non-uniform density of the Pareto front through its crowding distance mechanism.
3. **Exploration vs. Exploitation**
   Exploration: Achieved through mutation and crossover.
   Exploitation: Non-dominated sorting ensures convergence toward the Pareto front.
4. **Computational Efficiency**
   NSGA-II uses an $O(N^2)$ non-dominated sorting algorithm, which is computationally efficient even for large populations.

Advantages of NSGA-II for ZDT Problems

1. **Maintains Diversity:**
   Crowding distance prevents premature convergence to specific regions of the Pareto front.
2. **Scalable:**
   Works well with high-dimensional problems like ZDT4 and ZDT6.
3. **Handles Complex Fronts:**
   Discontinuous (ZDT3), non-convex (ZDT2), and multimodal (ZDT4) fronts are efficiently solved.
4. **No Need for Explicit Scalarization:**
   NSGA-II optimizes all objectives simultaneously without weighting.

Output of NSGA-II

After optimization, NSGA-II produces:

1. **Pareto-Optimal Front:**
   A set of solutions approximating the true Pareto front for the given problem.
2. **Diversity:**
   A well-distributed set of solutions across the Pareto front.

```
from pymoo.algorithms.moo.nsga2 import NSGA2
from pymoo.problems import get_problem
from pymoo.optimize import minimize
import matplotlib.pyplot as plt

# Function to solve a problem
def solve_problem(problem_name, n_var):
   # Get the problem
   problem = get_problem(problem_name, n_var=n_var)
   # Set up the NSGA-II algorithm
   algorithm = NSGA2(pop_size=100)
   # Perform optimization (limiting to 25000 evaluations)
   res = minimize(
       problem,
       algorithm,
       ('n_eval', 25000),  # Stopping criteria based on evaluations
       seed=1,
       verbose=True)

   return res.F
```

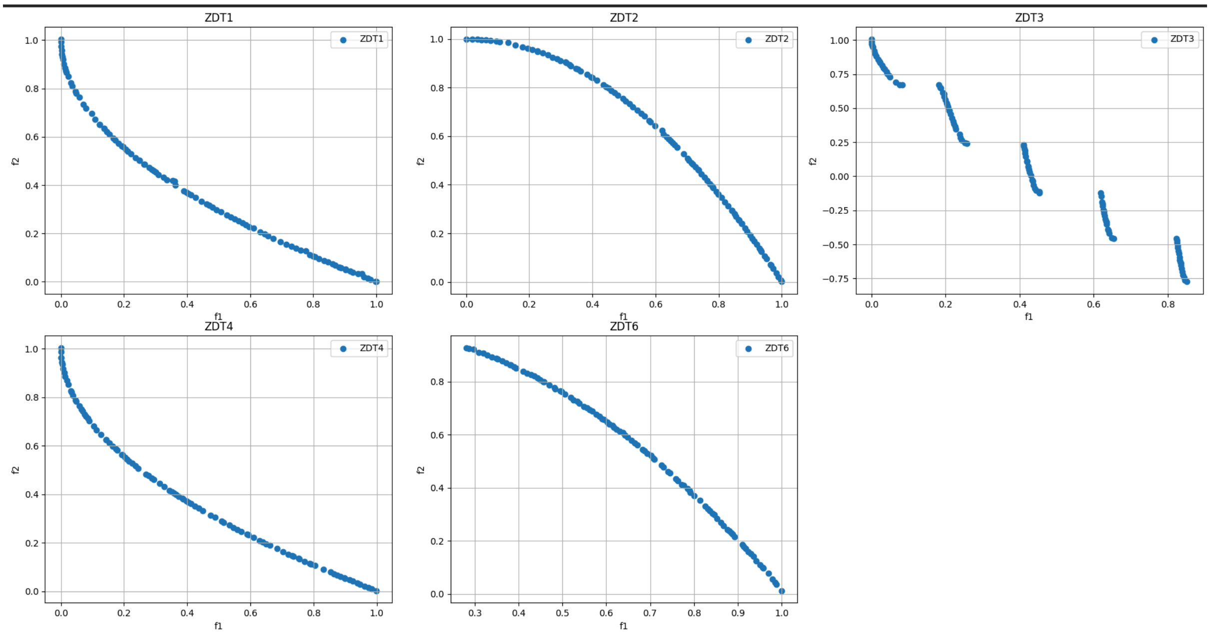
```python
# Solve all supported ZDT problems
zdt_results = {
    "ZDT1": solve_problem("zdt1", n_var=30),  # ZDT1 with 30 variables
    "ZDT2": solve_problem("zdt2", n_var=30),  # ZDT2 with 30 variables
    "ZDT3": solve_problem("zdt3", n_var=30),  # ZDT3 with 30 variables
    "ZDT4": solve_problem("zdt4", n_var=10),  # ZDT4 with 10 variables
    "ZDT6": solve_problem("zdt6", n_var=10),  # ZDT6 with 10 variables
}

# Plot results in subplots
fig, axes = plt.subplots(2, 3, figsize=(18, 12))

# Iterate through the ZDT results and plot
for i, (zdt_name, results) in enumerate(zdt_results.items()):
    ax = axes[i // 3, i % 3]
    ax.scatter(results[:, 0], results[:, 1], label=zdt_name)
    ax.set_title(zdt_name)
    ax.set_xlabel("f1")
    ax.set_ylabel("f2")
    ax.legend()
    ax.grid()

# Remove empty subplot if present
axes[1, 2].axis("off")

# Adjust layout
plt.tight_layout()
plt.show()
```



The plot visually represents the Pareto fronts generated for the five ZDT problems (ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6) after being solved with the NSGA-II algorithm.

1. **ZDT1 (Top-left)**
   Pareto Front: Convex.
   Explanation: The plot shows a smooth, convex curve where f_2 decreases as f_1 increases. NSGA-II has successfully captured the evenly distributed solutions on the front.
2. **ZDT2 (Top-middle)**
   Pareto Front: Non-convex.
   Explanation: The curve is slightly steeper compared to ZDT1 due to its non-convex nature. The algorithm captures the shape of the front, demonstrating its ability to handle non-convex problems.
3. **ZDT3 (Top-right)**
   Pareto Front: Disconnected (multiple segments).
   Explanation: The plot shows several disjoint segments, reflecting the problem's discontinuous nature. NSGA-II has successfully identified and maintained diversity across all segments of the Pareto front.
4. **ZDT4 (Bottom-left)**
   Pareto Front: Convex with multimodal decision space.
   Explanation: Although the Pareto front is similar in shape to ZDT1, ZDT4 includes 219 local optima due to multimodality. NSGA-II has converged to the true Pareto front, showcasing its robustness in multimodal landscapes.
5. **ZDT6 (Bottom-right)**
   Pareto Front: Non-convex and non-uniformly distributed.
   Explanation: The plot shows a steep initial slope near $f_1 = 0.3$, with solutions becoming sparse at $f_1 \to 1$. NSGA-II has captured the front's shape and non-uniform density, reflecting its ability to handle challenging distributions.

The plots confirm that:

Convergence: NSGA-II can find solutions close to the true Pareto front for all five ZDT problems.

Diversity: The solutions are well-distributed along the front, including disconnected (ZDT3) and non-uniform (ZDT6) cases.