Project MOO (Multi-Objective Optimization)

IA 2025-2026

Sonia YASSA

1. Problem Selection

- Choose a relevant problem in **IoT/Edge/Fog/Cloud Computing Environments** that requires multi-objective optimization. Potential problems:
    - **Workflow scheduling:** Optimize execution time, cost, latency… for edge/fog/cloud computing workflows scheduling.
    - **Task offloading:** Balance energy consumption, latency… for IoT tasks offloading to fog/cloud servers.
    - **Resource management:** Optimize resource allocation across cloud and edge nodes for minimal cost and energy usage…
    - **Optimization of electric vehicle (EV) charging:** Balance charging time, energy cost, and grid stability.
    - Or any other relevant real-world problem in the same ecosystem

2. Mathematical Formulation

- Define the problem using:
    - **Decision variables:** For example, in task scheduling, the decision variable could represent which node executes which task.
    - **Constraints:** These could include resource capacities, latency limits, budget constraints, etc.
    - **Objective functions:** Include at least **two objectives** (e.g., minimize execution cost, minimize latency, minimize energy consumption).
    - Use mathematical notation to express these clearly.

3. Metaheuristic Selection

- Pick a **Pareto-based approach** metaheuristic for multi-objective optimization. exemple algorithms:
    - Firefly Algorithm (FA): Known for swarm behavior.
    - Harmony Search (HS): Mimics the improvisation process of musicians.
    - Ant Colony Optimization (ACO): Based on pheromone trails of ants.
    - Bat Algorithm, Cuckoo Search (CS), Dragonfly Algorithm, Bacterial Foraging Algorithm (BFOA), Levy flight, …
    - Hybrid approaches: Combine metaheuristics with machine learning (e.g., Neural Networks for fitness prediction).

4. Implementation of the Metaheuristic

- Code the chosen metaheuristic, ensuring it supports **Pareto optimization**.
- Utilize a programming language like **Python**, **Java**,...
- Simulation environment: Simulator can be used:
  - Python-based simulators:
    - https://github.com/EdgeComputeSimSet/RayCloudSim
    - https://github.com/Cloudslab/FogBus2
    - https://github.com/Cloudslab/FLight
  - Java-based simulators
    - https://github.com/Cloudslab/cloudsim
    - https://github.com/Cloudslab/iFogSim
    - https://github.com/ISEC-AHU/FogWorkflowSim

5. Performance Evaluation

- Compare your metaheuristic's results with:
  - Other existing metaheuristics.
  - Baseline algorithms (like random selection, greedy algorithms).
- Use metrics such as:
  - **Hypervolume indicator:** Measures the Pareto front's quality.
  - **Spacing metric:** Evaluates the distribution of solutions.
  - **Pareto front visualizations:** 2D or 3D graphs.

6. Report and Presentation

- **Report:** Include the following:
  - Problem introduction and importance.
  - Mathematical formulation with decision variables, constraints, and objectives.
  - Metaheuristic adaptation to the problem.
  - Implementation details (flowchart, pseudocode).
  - Performance analysis and comparison.
  - Insights and conclusions.
- **Presentation:** A concise, engaging slide deck explaining:
  - Problem and its significance.
  - Mathematical modeling and chosen metaheuristic.
  - Key results and visualizations.
  - Demonstration of the application.

7. Proposed Timeline	**Group B**	**Group A**
Problem selection	09/11/2025	16/10/2025
Problem modeling and Metaheuristic choice	16/11/2025	19/11/2025
Report and presentation	11/12/2025	20/01/2026