Configuration Manual

MSc Research Project
Cloud Computing

Myungjee Koh
Student ID: x23268409

School of Computing
National College of Ireland

Supervisor: Shaguna Gupta

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Myungjee Koh
Student ID:	x23268409
Programme:	Cloud Computing
Year:	2025
Module:	MSc Research Project
Supervisor:	Shaguna Gupta
Submission Due Date:	11/08/2025
Project Title:	Configuration Manual
Word Count:	634
Page Count:	5

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	13th August 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Myungjee Koh
x23268409

This paper provides a detailed guide for replicating the experimental setup used in the proposed Pareto-based Deep Q-Network (PDQN) load balancing framework.

1 Environment Setup

- **Operating System:** The simulation tested on macOS - Apple M1
- **Python Version:** 3.9.7 (installed via `pyenv`). Due to simulator.
- **Virtual Environment:** Created using `pyenv virtualenv yafs_env`.
- **Key Libraries:**
  - YAFS (Yet Another Fog Simulator)
  - TensorFlow 2.15 for implementing the PDQN agent.
  - Pandas, NumPy for data handling and CSV processing.
  - Matplotlib or plotting simulation results and Pareto front visualizations.
- All library are defined in `requirements.txt`

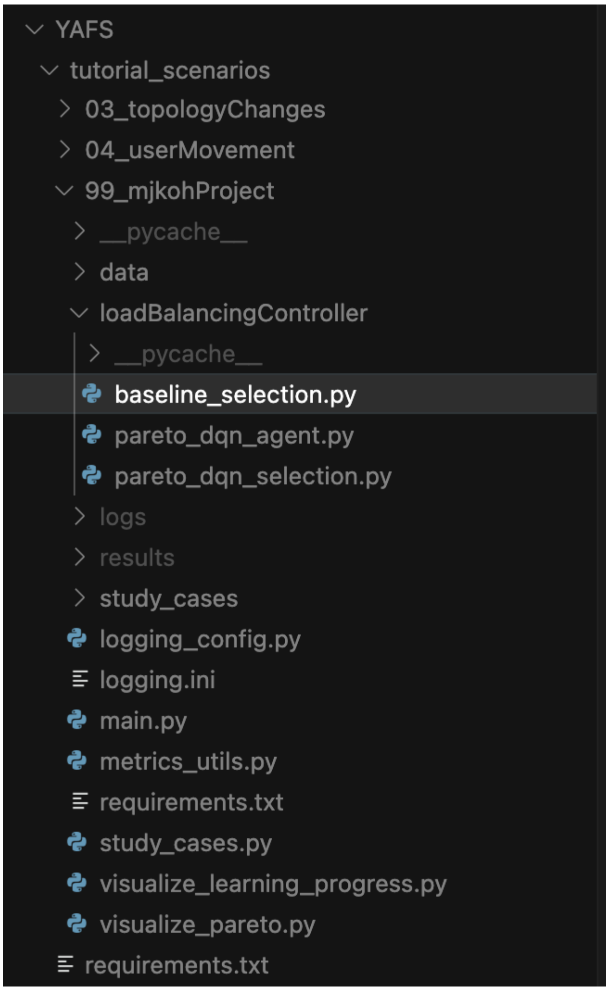
Example installation commands:

```
pyenv install 3.9.7
pyenv virtualenv 3.9.7 yafs_env
pyenv activate yafs_env
pip install -r requirements.txt
```

2 Project Structure

- YASF/tutorial_scenarios/99_mjkohProject — The proposed PDQN load balancing algorithm is built in this root directory by extending YAFS simulator's scenarios.
- loadBalancingController/ — In this directory, every load balancing approach are implemented including baseline alogithm and propoesed PDQN.
- `pareto_dqn_agent.py` — Implementation of the Pareto-based DQN agent.
- `pareto_dqn_selection.py` — Decision-making and routing logic followed the YAFS simulation's flow.

- `metrics_utils.py` — Real-time metrics collection, calculation and normalization latency and energy consumption, and common utility.
- `topologyDefinition.json` — Network topology configuration.
- `usersDefinition.json` — Source workload definitions.
- `results/` — Simulation outputs (CSV files, summary tables, visualization).
- `main.py` — control entire simulation such as setting topology, app, and deploying the simulation.



```
YAFS
  tutorial_scenarios
    03_topologyChanges
    04_userMovement
    99_mjkohProject
      __pycache__
      data
      loadBalancingController
        __pycache__
        baseline_selection.py
        pareto_dqn_agent.py
        pareto_dqn_selection.py
      logs
      results
      study_cases
      logging_config.py
      logging.ini
      main.py
      metrics_utils.py
      requirements.txt
      study_cases.py
      visualize_learning_progress.py
      visualize_pareto.py
    requirements.txt
```

Figure 1: project structure

3 Simulation Configuration

- Three study cases are defined in `study_cases.py`.
  It creates `study_info.json`, `allocDefinition.json`, `topologyDefinition.json` for each study cases. `topologyDefinition.json` includes entity id, model (name), IPT, RAM, cost, and WATT. WATT is used for energy consumption calculation.
  - **Small Scale:** 10 fog nodes

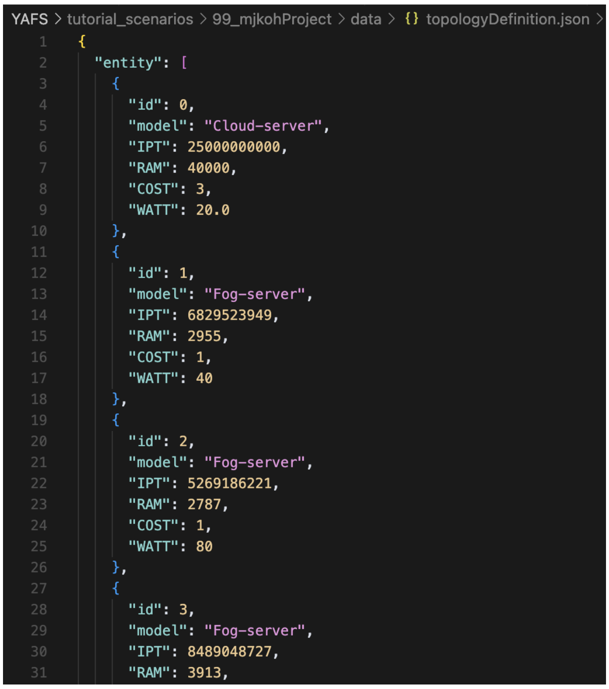
  - **Medium Scale:** 20 fog nodes
  - **Large Scale:** 30 fog nodes

- The `StudyCase` class in the configuration script allows adjusting:
  - Fog CPU range
  - Bandwidth range
  - Number of tasks per IoT device

- Simulation time can be set in the main experiment script to control the training length.

- `allocDefinition.json` includes module_name, app, id_resource. It defines modules deployment.

- `appDefinition.json` defines module, message, transmission etc.


YAFS > tutorial_scenarios > 99_mjkohProject > data > {} topologyDefinition.json >

```
{
  "entity": [
    {
      "id": 0,
      "model": "Cloud-server",
      "IPT": 25000000000,
      "RAM": 40000,
      "COST": 3,
      "WATT": 20.0
    },
    {
      "id": 1,
      "model": "Fog-server",
      "IPT": 6829523949,
      "RAM": 2955,
      "COST": 1,
      "WATT": 40
    },
    {
      "id": 2,
      "model": "Fog-server",
      "IPT": 5269186221,
      "RAM": 2787,
      "COST": 1,
      "WATT": 80
    },
    {
      "id": 3,
      "model": "Fog-server",
      "IPT": 8489048727,
      "RAM": 3913,
```

Figure 2: example of topologyDefinition.json

4 Running the Experiments

1. Ensure the virtual environment is activated:

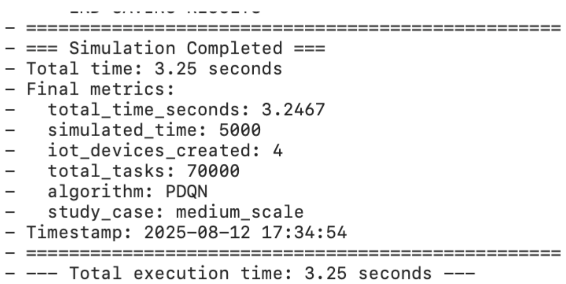
   ```
   pyenv activate yafs_env
   ```

2. Select the desired scale in the experiment configuration.

3. Run the simulation:

```
python3 main.py -t 5000 -s small_scale -a ALL
```

4. -t : simulation time
5. -s : study case (small_scale, medium_scale, large_scale)
6. -a : algorithm (RR, WRR, PDQN, ALL)
7. Results will be stored in `results/<scale>`.



```
- ==================================================
- === Simulation Completed ===
- Total time: 3.25 seconds
- Final metrics:
-   total_time_seconds: 3.2467
-   simulated_time: 5000
-   iot_devices_created: 4
-   total_tasks: 70000
-   algorithm: PDQN
-   study_case: medium_scale
- Timestamp: 2025-08-12 17:34:54
- ==================================================
- --- Total execution time: 3.25 seconds ---
```

Figure 3: simulation log example

5 Notes for Reproducibility

- The cloud server (node ID 0) is excluded from load balancing decisions.
- All routing decisions follow YAFS's `__send_message()` mechanism to simulate realistic network delays.
- Users may adjust the `lambda` parameter in `usersDefinition.json` to control the message generation rate.
- Hyperparameters such as learning rate, epsilon decay, and reward normalization are adjustable in `pareto_dqn_agent.py`.

6 Output and Visualization

- **CSV Outputs:** Every output is stored in results directory. In this directory, outputs are stored divided by study cases.
  - `result/small_scale/link_PDQN.csv`, `result/small_scale/result_RR.csv`, etc.
  - `result/medium_scale/simtrace_PDQN.csv`, etc.
  - `result/medium_scale/pareto_front_PDQN.csv` : recorded Pareto front values per episode.
  - `result/small_pareto_summary.csv`, etc.: latency/energy trade-off summary.
- **Graph:**

  - `resutl/<scale>/sim_trace_<algorithm>_grahp.gexf` — They are created by YAFS simulator defined every topology network in the graph.

- **Plots:**

  - latency, energy consumption comparison to each algorithm per scale.
  - pareto front, learning progress per episode for PDQN algoirthm. These two scatter plots did not use in the report because of ambiguous to show the improvement of learning.

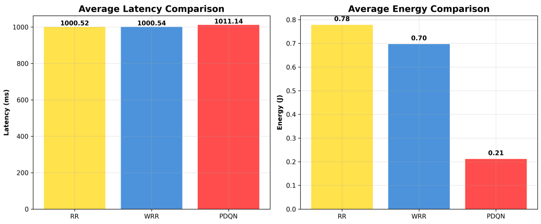
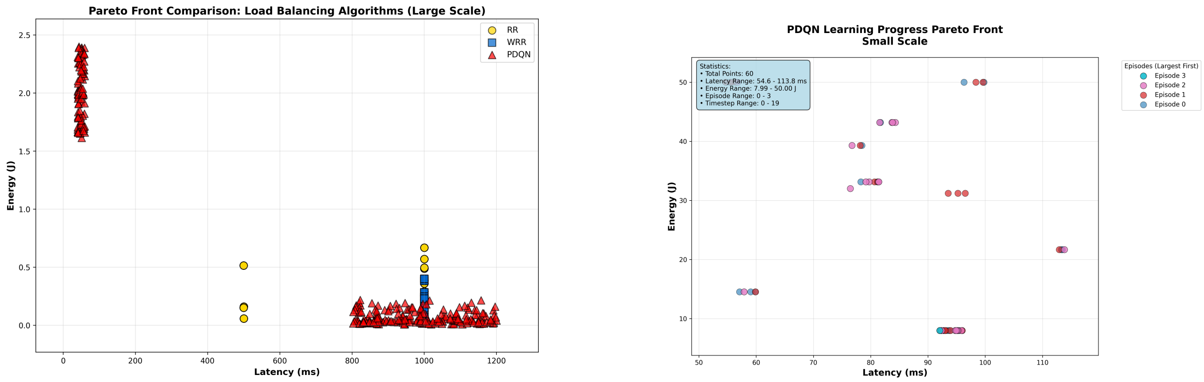


Figure 4: small scale algorithm comparison



(a) large scale pareto_front comparison

(b) pareto front learning progress

Figure 5: These plots did not use in the report