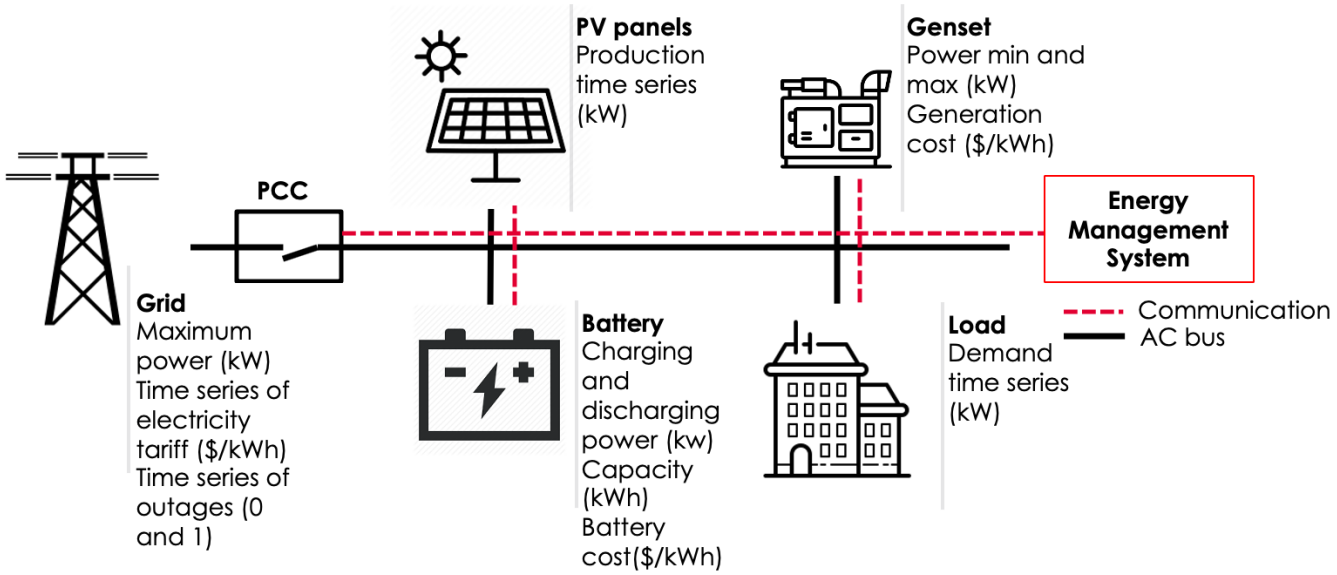
pymgrid: An Open-Source Python Microgrid Simulator for Applied Artificial Intelligence Research

Gonzague Henri, Tanguy Levent, Avishai Halev, Reda Alami, Philippe Cordier

12/11/2020

https://github.com/Total-RD/pymgrid

What is pymgrid?

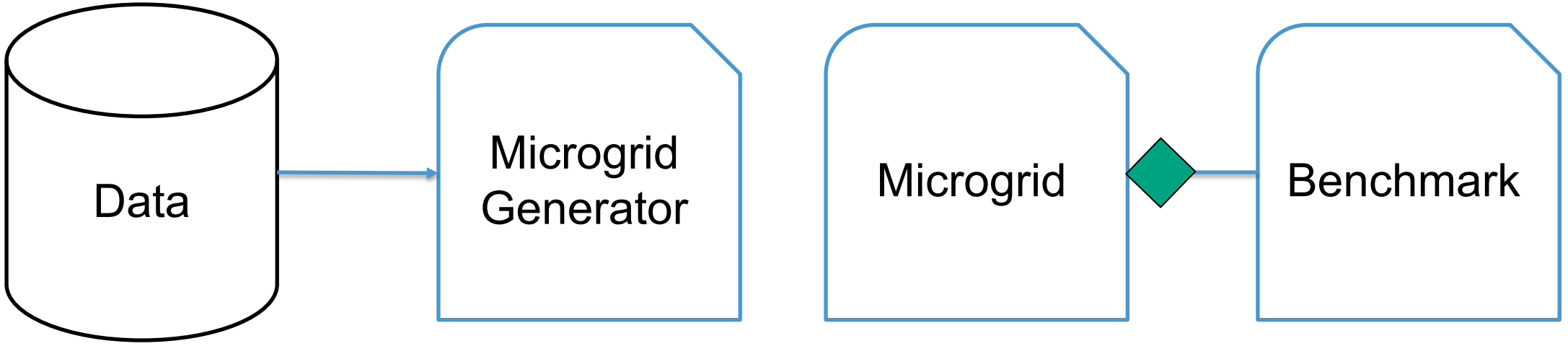


pymgrid is an open-source microgrid simulator. It has been built to be modular and for applied RL research

pymgrid materialized from the observations:

- Lack of open data for microgrids
- There was no way to generate a large number of microgrids for AI research
- Existing simulators did not have RL-friendly API

Architecture: one class to generate microgrid, one to handle the simulation



- Load, PV and CO2 time series
- 1-year hourly data

- Uses data files to generate microgrid objects
- Smart sizing
- Random microgrid architecture
- Can generate 600+ different microgrids

- Class to interact with during simulation
- Has some plotting/utility functions
- Implement control loop and constraint verification

- Class to compute benchmarking algorithms
- Can also generate forecasts for said algorithms

Code to interact with microgrid

```
#With train_test_split, the microgrid data is split into a training and a
    testing set
mg.train_test_split()

#an agent can be trained with a loop based on the one below:
for epoch in range(nb_epoch): # that needs to be specified by the user
    mg.reset()
    while not mg.done: #done turns to True at the last timestep
        ctrl = #an action to fill the control_dict
        mg_data = mg.run(ctrl)

#now that the agent is trained, we can switch to the testing set
mg.reset(testing=True)

#we can use the same loop to go through the testing set
while not mg.done:
    ctrl = #
    mg_data = mg.run(ctrl)
#we can print the final cost:
mg.print_cost()
```

Function to reset the time series dictionaries and reset the battery

Move forward one time-step

Passing testing=true change the dataset used to the testing set

Done = True indicates the last timestep has been reached

Scenarios: two datasets that can be used to test control algorithms

Table 4: Numerical results on `pymgrid25`

Architecture	Metric (k$)	MPC	Rule-based	Q-learning	Q-learning + DT
All	Mean cost	11,643	19,265	389,234	**13,385**
	Total cost	291,086	481,636	9,730,870	**334,624**
Genset only	Mean cost	19,722	57,398	337,385	**24,777**
	Total cost	78,890	229,593	1,349,543	**99,109**
Grid only	Mean cost	8,150	**8,372**	383,105	8,524
	Total cost	73,352	**75,350**	3,447,945	76,718
Grid + Genset	Mean cost	19,107	**22,327**	480,107	22,376
	Total cost	57,322	**66,982**	1,440,322	67,130
Weak grid	Mean cost	9,058	12,190	388,118	**10,185**
	Total cost	81,522	109,711	3,493,059	**91,666**

- We would like to generate scenarios of pre-computed microgrids
  - **Pymgrid25**: microgrids with different architecture
- In those scenarios, the benchmark algorithms would already be computed, can be used to compare performance across algorithms / research group

Two Gym environments are available

Gym environments

- The environment follow standard gym API, taking one microgrid as ”env_config”
- Notebook available for an example how to use the environment with Ray rllib
- So far, two environment with discrete action spaces are available:

Priority list

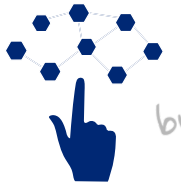
Discretized action space

Code available on github:

https://github.com/Total-RD/pymgrid

You can install it today with

pip install git+https://github.com/Total-RD/pymgrid/



by Computational Science Engineering
Data Science & Artificial Intelligence