Job interview exercise

Agregio Solutions / Direction ingénierie informatique industrielle

May, 29th 2023

Problem description

An industrial site is connected to the national electrical grid, later referred to as “the Grid”. Within the industrial site, electricity distribution is managed by a smart grid.

> You have to develop the distribution algorithm of an EMS[1] that controls this industrial site’s smart grid. You must translate the system constraints into equations and statements for the different devices.

On the industrial site smart grid, three devices are connected to a common AC bus[2]:

- the industrial facility load
- an Energy Storage System (ESS, e.g. a battery) of capacity `ess_capacity` in kWh
- a photovoltaic (PV) power plant of peak power `pv_peak` in kW

The EMS can read the power `Pess` flowing from/to the ESS, and send power setpoint to the ESS (charge/discharge setpoint).

The PV power plant produces electricity, and cannot be controlled. The EMS can read the power `Ppv` flowing from the PV power plant.

The industrial facility load consumes electricity, and cannot be controlled. There is no direct metering of industrial facility power consumption `Pload`, but it can be deduced by the EMS.

At the point of connection (POC) of the AC bus with the Grid, a fourth device is installed:

- a meter measures `Ppoc`: the power flowing from/to the Grid.

`Ppoc` is the resultant of smart grid productions and consumptions:
`Ppoc = Pess + Ppv + Pload`. The EMS can read `Ppoc`, but remember that `Pload` is not directly available for the EMS.

> The EMS objective is to ensure that the industrial site power consumption remains under a maximal value `PmaxSite`. This means, the EMS must keep `PmaxSite < Ppoc` at all times. Exporting power to the Grid is tolerated.

[1] **EMS**: Energy Management System

[2] **AC Bus**: An AC bus is a central point in an electrical system where multiple power sources and loads are connected. It serves as a distribution point for alternating current (AC) power within the network. The AC bus ensures that the power is distributed among the connected loads and maintains the stability of the electrical network.

Sign convention used

- `P < 0` for power consumption
- `P > 0` for power production

Variables description

All variables are measured on the “AC side” of the relevant energy conversion devices.

ESS variables

Read-only values

- `Pess`: current ESS active power output in kW (`< 0` means charge / `> 0` means discharge)
- `Pmaxch`: current ESS maximal charge power capability in kW (necessarily `<= 0` by convention)
- `Pmaxdisch`: current ESS maximal discharge power capability in kW (necessarily `>= 0` by convention)
- `Eess`: current ESS stored energy in kWh (necessarily `>= 0`)

Write values

- `setpointPess`: active power setpoint computed by the EMS in kW (`< 0` for charge setpoint, `> 0` for discharge setpoint)

PV variables

Read-only values

- `Ppv`: current PV inverter active power output in W (necessarily `>= 0` by convention)

POC meter variables

Read-only values

- `Ppoc`: current active power measure at POC in kW (`< 0` means smart grid imports power from the Grid, `> 0` means smart grid exports power to the Grid). We expect `Ppoc` to comply with `PmaxSite < Ppoc` rule.

Site load variables

Read-only values

- `Pload`: current industrial site power consumption in kW (`<= 0` by convention)

Snapshot example

The following set of variables illustrates an example:

- `Pess = 120 kW` : ESS discharging, with power equal to 120 kW
- `Ppv = 200 000 W` : PV power plant producing, with power equal to 200 kW
- `Ppoc = -300 kW` : industrial smart grid drawing power from the Grid, with power equal to 300 kW which would correspond to a current industrial site consumption of 620 kW (`Pload = -620 kW`)

Constraints and expectations

You have at your disposal a set of functions that return raw measures of the system (assume values directly come from the devices). The sign and unit for the measure variables are defined above.

- `GetEssMeasure()` returns `Pess, Pmaxch, Pmaxdisch, Eess`
- `GetPvMeasure()` returns `Ppv`
- `GetPocMeterMeasure()` returns `Ppoc`

Also, you have a set of functions to send setpoints to devices EMS can control:

- `SetEssSetpoint(setpointPess)` sends `setpointPess` to ESS

You can use the **development language of your choice**.

Please note we do not focus on language syntax here, we also **do not** expect the solution you write to be executable, or having a program with hundreds of lines. Writing a solution with pseudo-code is totally acceptable.

Also keep in mind that this is an open problem: there is no expected solution.

The key point for us is trying to get a global idea on how you address such a problem, how you present the solution, on what details you choose to focus. So, don’t hesitate:

- to add any comment that could help us to have a better understanding on what assumptions you make,
- to make use of functions that you do not implement, as long as you describe its interface and its behavior.