PPC project

The Energy Market

Razmig Kéchichian, Jilles Dibangoye & Frédéric Le Mouël

1. Goal

The goal of this programming project is to design and implement a multi-process and multi-thread simulation in Python. This document defines the functionalities that it has to implement at a minimum. Any extensions and developments will act in your favor in the evaluation. In addition, some points are deliberately left open for you to propose your own solution. Be rigorous and creative!

2. Presentation

2.1. Minimum specifications

The program simulates an energy market where energy-producing and consuming homes, weather conditions and random events contribute to the evolution of energy price over time. Homes may choose to give away their surplus of energy to other homes that are in shortage, or may sell it to the market thus contributing to the diminution of energy price. Homes in need of energy will have to buy it from the market if others would not give away any. Naturally, energy prices climb when the average consumption exceeds that of production. Temperature changes also impact energy consumption, hence its price. Other events, such as the enactment of a law, diplomatic tension etc. can impact energy prices.

2.2. A minimal design

The simulation involves 5 types of processes.

- `home`: represents an energy-producing and consuming home, with initial production and consumption rates and energy trade policy which is one of three possibilities 1. always give away, 2. always sell on the market, and 3. sell if no takers.
- `market`: represents the market with current energy price which evolves according to transactions with homes (selling and buying), weather conditions and other events. The market process is multi-threaded and carries out transactions with homes in separate threads. Furthermore it limits the number of simultaneous transactions that can take place with homes.
- `weather`: simulates weather parameters impacting energy consumption, such as temperature and its variation over time.
- `politics`: simulates political events parameters impacting energy consumption, such as diplomatic tension, wars.
- `economics`: simulates economic parameters of predefined types that can impact the energy price, such as money change rate, world macro-economy, carbon price.

Inter-process communication: Processes in the simulation communicate in the following way. `home` processes communicate among themselves and with the `market` via message queues. The `weather` process updates a shared memory with weather condition parameters. The `politics` and `economics` processes, childs of `market` process, signals events to the latter which takes the corresponding action impacting energy price. `home` and `market` processes update terminals

they are connected to permitting the operator of the simulation to track its progress.
Energy price: The energy price can be calculated according to the following linear model:

$$P_t = \gamma P_{t-1} + \sum_i \alpha_i f_{i,t} + \sum_j \beta_j u_{j,t}$$

P_t	is the energy price at instant t	€/kWh
P_{t-1}	is the energy price at instant $t-1$	€/kWh
γ	is a long-term attenuation coefficient	unitless
$f_{i,t}$	is the contribution of the i^{th} internal factor at instant t, such as the current temperature, amount of energy sold or bought on the market etc.	depends on $f_{i,t}$
α_i	is a modulating coefficient for factor $f_{i,t}$	€/kWh
$u_{j,t}$	is the contribution of the j^{th} external factor at instant t, such diplomatic tension, social unrest, fuel shortage etc. To simplify, we consider that $u_{j,t} \in \{0,1\}$ indicating the absence/presence of a given external circumstance	unitless
β_i	is a modulating coefficient for factor $u_{j,t}$	€/kWh

Here's a numeric example. For $\gamma = 0.99$, $P_{t-1} = 0.145$, $\alpha_0 = 0.001$, $f_{0,t} = 1/5°$ the reverse of current temperature, $\beta_0 = 0.01$, $u_{0,t} = 1$ presence of fuel shortage, the price of energy at instant t is $P_t = 0.15375$. In your design, you are required to define at least 2 internal and 3 external factors.

3. Solution

3.1. Design

Start with a diagram to better visualize the interactions between processes. The following points need to be defined:

- relationships between processes (parent-child or unrelated)
- messages exchanged between processes along with their types
- data structures stored in shared memory and how they are accessed
- synchronization primitives to protect access to shared resources or to count resources
- signals exchanged between processes, if any, and their types
- tubes involved in pairwise process communication, if any, and their types

Write down a Python-like pseudo-code of main algorithms for each process.

3.2. Implementation plan

Plan your implementation and test steps. We strongly encourage you to first implement and test every process separately on hard-coded data in the beginning. Then you can implement and test each pair of communicating processes separately from other processes, again on hard-coded data if necessary. Only after implementing and testing each inter-process communication you can put processes together and test the simulation. Think about how to automate the startup and the proper shutdown of the simulation, freeing all resources. Identify possible failure scenarios and think about recovery or termination strategies.