Parallel Programming – task 1

26.02.2017

Design and implement the synchronization protocol (based on semaphores) for the following problem.

Problem

1. Three factories produce Lead, Sulfur, and Mercury (one Factory for each resource). Alchemist use these resources to create gold. Additionally, evil Warlocks cast curses on Factories to prevent production, and good Sorcerers remove the curses.
2. If a Factory is not cursed (i.e., the number of curses on this factory is 0), it produces its resource and stacks it. Producing one unit of a resource takes some random interval of time. If two units are already produces, the Factory waits (there is no more space in the storage). The Factory also waits if it is cursed.
3. A Warlock, in random intervals of time, casts a curse on a random Factory. The curses stack, i.e., if this Factory is already cursed, it is now cursed twice, and two curses have to be removed to make the Factory work again.
4. A Sorcerer, in random intervals of time, removes one curse from each cursed Factory.
5. There are four guilds of Alchemists trying to obtain gold: $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$.
6. Alchemists $\mathcal{A}$ try to obtain gold from lead and mercury.
7. Alchemists $\mathcal{B}$ try to obtain gold from mercury and sulfur.
8. Alchemists $\mathcal{C}$ try to obtain gold from lead and sulfur.
9. Alchemists $\mathcal{D}$ try to obtain gold from mercury, sulfur and lead.

10. Alchemists acquire the resources from Factories, they can take resources that are already in the storage. They come in random order. There is some (random) interval between the appearances of two Alchemists.
11. Each Alchemist tries to collect all necessary ingredients. If they are not available, the Alchemist waits.
12. After collecting all the ingredients, each Alchemist goes to his laboratory (and does not appear in the story anymore).

Technical aspects

1. Print the current state of the program on the console.
2. Each character (Factory, Wizard, Sorcerer, or Alchemist) is represented by one thread or process. Additional threads or processed may be used if necessary.
3. The number of Wizards and Sorcerers is a parameter, start with 3, but it should be possible to adjust it.
4. Choose some limits on time intervals mentioned in the task, so that we can see what is happening. It should be possible to change these limits easily. The synchronization protocol should not be based on these time limits!
5. The solution should offer the maximum parallelism. The following situation is erroneous:
   There is some mercury and sulfur available (but no lead). There are some Alchemists $\mathcal{B}$ waiting, but they are blocked by the Alchemist $\mathcal{D}$, who appeared earlier.
6. You should avoid the starvation of Alchemists $\mathcal{D}$.
7. The solution when the Alchemist takes some ingredients and returns them (if not everything he needs in available) is not the best possible (and therefore it is not worth maximum number of points).
8. Any kind of busy waiting (even a little one) is wrong.
9. The synchronization should use only simple semaphores and methods (in Java: `acquire()` and `release()`, in C#: `WaitOne()` and `Release()`, in other languages: if you are not sure, ask the teacher).

Deadline for submitting your solution: 19.03.2018, 07:00 AM.