Assignment 3: Bilevel Deterministic Problem

Submitted by: Babu Kumaran Nalini

Introduction:

A market clearing problem is chosen as an example for the deterministic problem which is solved using a bilevel programming. A python code is tested in two models namely a simply example as discussed in the lecture and with the IEEE RTS 24-bus system. The execution of the Python code results in the optimization results along with the complementarity condition check.

Problem Derivation:

In this example, for the bilevel programming the existing IEEE 24 bus system is considered and it is possible to include in the Python program the leader generators as list. Considering the leader, the Python program executes in order to find out the cost minimization of the market clearing problem.

To the simplify this report, the complete problem statement is not re-written. The Python code exactly implements the strategic offering problem conditions discussed made during lecture on bilevel programming Lecture 7, slide number 88.

Python Model:

- Input file
  - 2-bus system (based on lecture) / 24-bus IEEE data
  - Excel data input
- Specifications
  - Total constraints: >1000 for Stochastic model and >39 per real-time optimization
  - Solver: Gurobi using Pyomo
  - Processor: Core i7 8550
  - Total time: 18 seconds
- Output
  - Optimized market clearing cost
  - Complementarity condition check

Results

In the following section, a bilevel optimization problem is solved using Python. Here a deterministic case is considered. The input dataset for the following results is the IEEE RTS 24 bus system data. Here, generators 1 and 22 are considered to be the **leader**. Even though the IEEE 24 bus system gives a specific demand value in this example we have assumed some random demand pattern for the different demands used in the excel file. Along with the execution the complementarity condition for the leader

are checked in order to be sure that the solution is correct. Also, to linearize the complementarity conditions a Fortuny-Amat approach or the Big-M approach is used. Here, the constant M is assumed to be a very value which is equal to 10000.

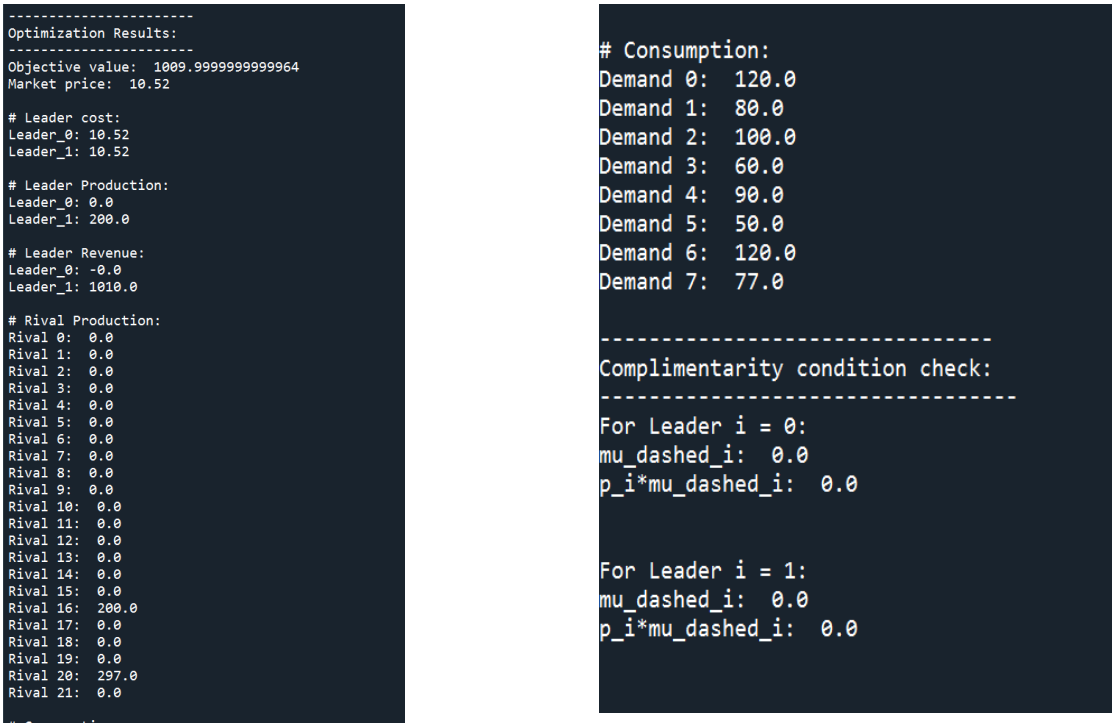

Figure 1 Snippet from python program execution

Figure 1 shows a snippet from the python program execution and the results of the bilevel program with the generation of each generators and the respective demands of the optimization problem.

Discussion

- The Python code can take in different sets of generators as leaders.
- It can be seen the market price is decided by the second cheapest generator as the first generator alone could not satisfy the demands.
- Even though there were two leaders the leader 1 chosen in this example was such that his value was actually not impacting the market. Therefore, another generator which offered cheaper generation served as a player.
- The complementarity condition is checked in order to confirm the optimization problem is solved correctly using the big M approach.
- Varying the big M value in a lower range varies the problem solution. Therefore, a relatively big value was chosen which didn't disturb the problem solution.
- In this case, a common big M was used as discussed in one of the examples during the lectures.

****