Homework Problem 2

Formal Methods in Robotics (Fall 2019)

In this problem, we want to implement the product of a Transition System (TS) and an Automaton (NFA). See Definition 4.16 from Principles of Model Checking by Baier and Katoen.

To complete this HW, implement the following functions in `prod_ts_aut.py`.

- `Game._define_by_tsys_aut`: Implement the product operation.
- `test_ts_aut_product`: Test whether your implementation is correct.

To write the test function, you will have to choose a transition system and an LTL specification for the robot. Describe the TS and specification you choose while submitting the HW code. Also, justify the test cases you used to check the correctness of your implementation.

Hint: If you are unfamiliar writing test cases, you might want to see `test_1x1_world` and `test_2x2_world` functions from `gw_graph.py`.

You will need to be familiar with following objects.

- Class Graph : See `iglsynth.util.graph` module
- Class TSys: See `iglsynth.game.tsys` module
- Class Automaton: See `iglsynth.logic.core` module
- Function AP: See `iglsynth.logic.core` module
- Class PL: See `iglsynth.logic.core` module
- Class Game: See `prod_ts_aut.py`

Pay attention to `Automaton.Edge` that is defined as the tuple (u, v, ϕ), where ϕ is a propositional logic formula. The label of a state $s \in TS$ can be evaluated by running `Alphabet(<TSysObject>.props).evaluate(s)` which returns a dictionary of `{key=AP: value:bool}`.

Also, an edge of transition system `TSys.Edge` is defined as a tuple (u, v, a), where a is an action.