Missouri University of Science & Technology — Department of Computer Science
**Fall 2023** — **CS 5408: Game Theory for Computing**

# Project 2

**Instructor:** *Sid Nadendla* — **Due:** *December 5, 2023*

**Instructions:** Students who did not follow any of the following instructions will be ignored and a zero grade will be rewarded accordingly.

- The main goal of this assignment is to implement a Python package `gtclab` for representing and solving both normal and extensive games from scratch.
- You are **not** allowed to import any other Python library, other than the ones that are already imported in the code-base.
- You are also **not** allowed to add, move, or remove any files, or even modify their names.
- You are also **not** allowed to change the signature (list of input attributes) of each function.

## Problem 0 Extensive Game Representation *10 pts.*

Copy all the code you wrote for Project 1, that supports representation of games in extensive form, into their respective locations in Project 2. These files include:

- project1/gtc-lab/base/state.py $\Rightarrow$ project2/gtc-lab/base/state.py
- project1/gtc-lab/base/tree.py $\Rightarrow$ project2/gtc-lab/base/tree.py
- project1/gtc-lab/models/extensivegame.py $\Rightarrow$ project2/gtc-lab/models/extensivegame.py

## Problem 1 Subgame Perfect Equilibrium *60 pts.*

Implement each of the following classes and methods listed below, which can be found in project2/gtc-lab/solvers/spne.py:

- `spne()`: This class solves any perfect-information extensive game using the notion of *subgame perfect equilibrium.* In this algorithm, the value of a given state is the Therefore, define the following four static methods accordingly:
  - `set_state_value()`: Set the value of the state, whose label is given.
  - `get_state_value()`: Get the value of the state, whose label is given.
  - `set_subtree_equilibrium()`: Set the equilibrium of the subtree rooted at the state, whose label is given.

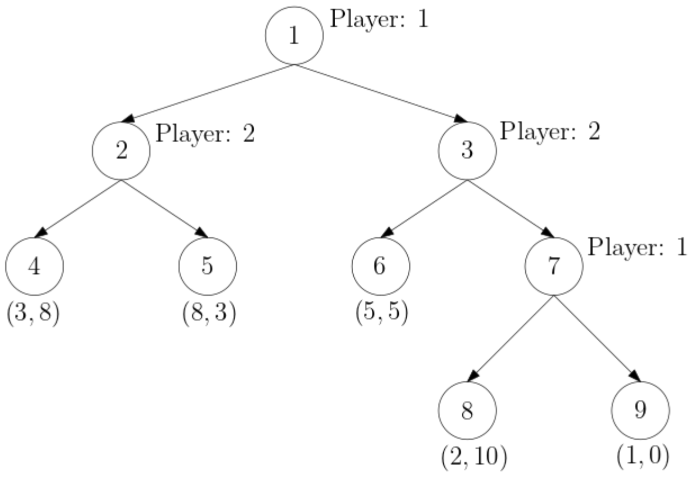

Figure 1: Extensive Game for Problem 3(a)

- `get_subtree_equilibrium()`: Get the equilibrium of the subtree rooted at the state, whose label is given.
- `find_subtree_NE()`: Find the Nash equilibrium of the subtree rooted at the state, whose label is given.

## Problem 2 Validation ***30 pts.***

Retrieve the Jupyter notebook from Project 1 with the implementation of the extensive game shown in Figure 1. Solve this game using your own `spne` solver.