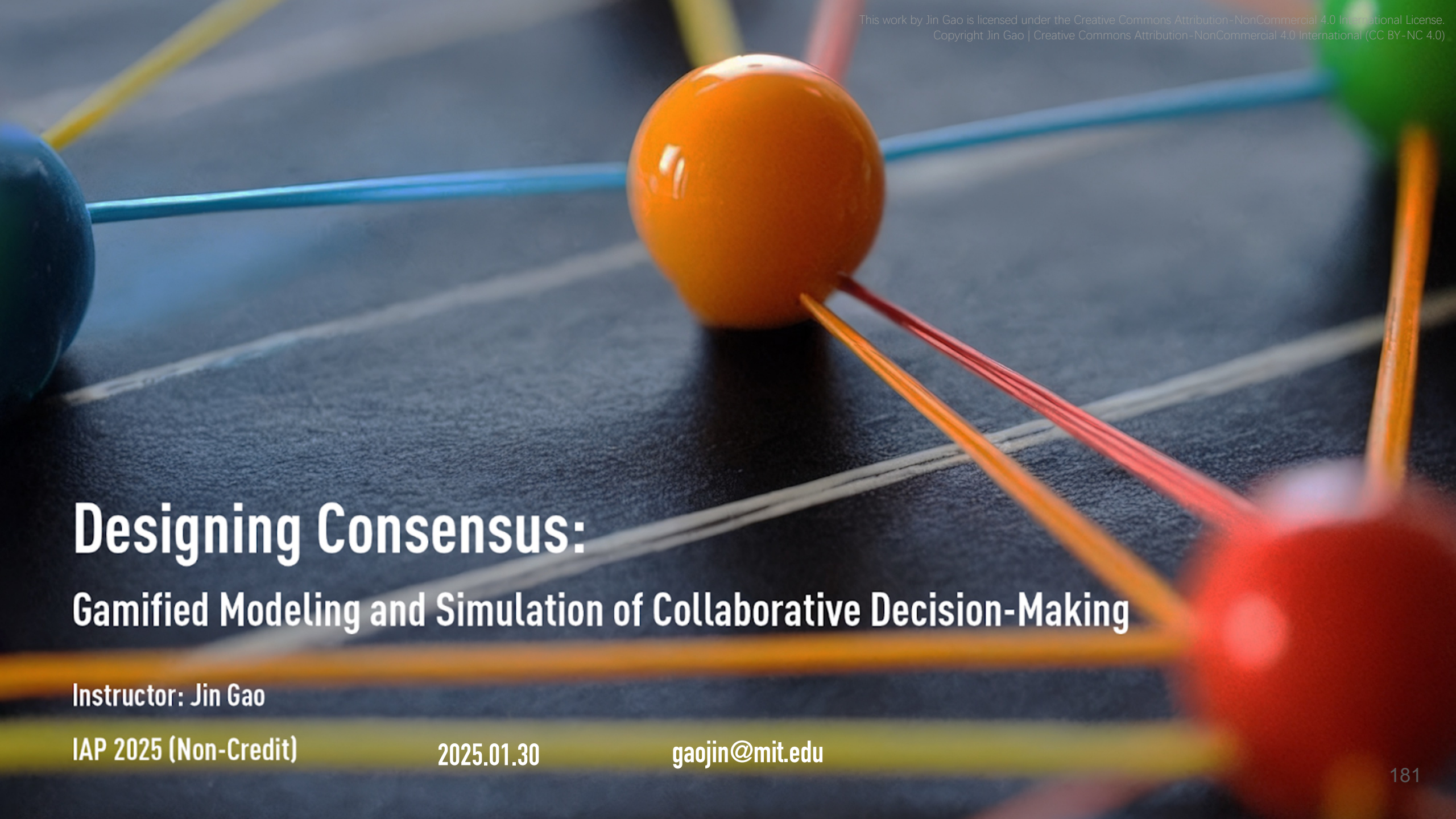


Designing Consensus:

Gamified Modeling and Simulation of Collaborative Decision-Making

Instructor: Jin Gao

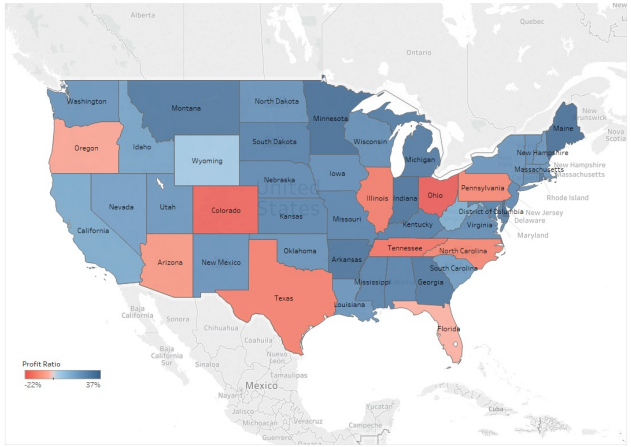
IAP 2025 (Non-Credit) 2025.01.30 gaojin@mit.edu

Date	Topic	Contents
Jan 30	Multi-Agent Learning	● Step from the concept of game theories we built from the previous lectures, we explore methods to train agents to make decisions in the environment we created. Starting from reinforcement learning (RL), we will introduce multi-agent RL and explore the pros and cons of a variety of methods. ● Overview of advances in this field, including Deep Blue, AlphaGo, and Cicero.
	Workshop	● Present your final game design to the class (game rules, competition or collaboration mechanisms, balancing, strategies, etc.).
	Wrap up and conclude.	

Before making a decision, what info do we need?

Descriptive

What happened?

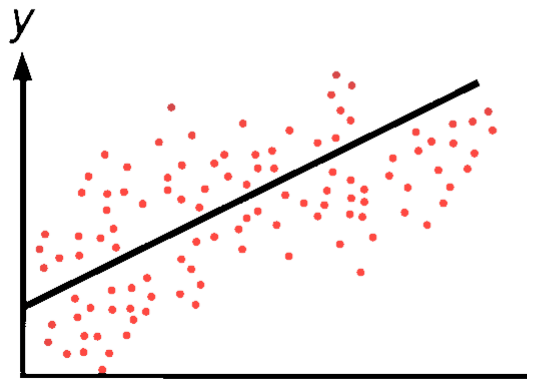


Data Management

Data Visualization

Predictive

What will happen?

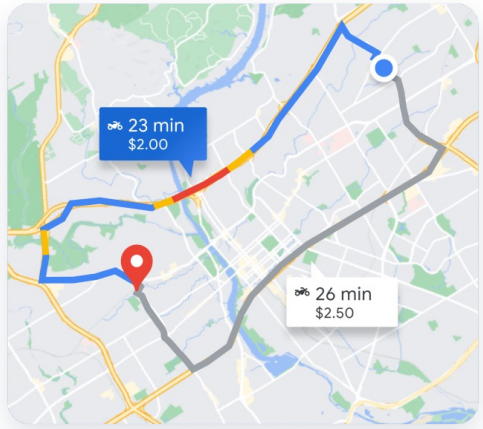


Forecasting

Machine Learning

Prescriptive

What should I do?



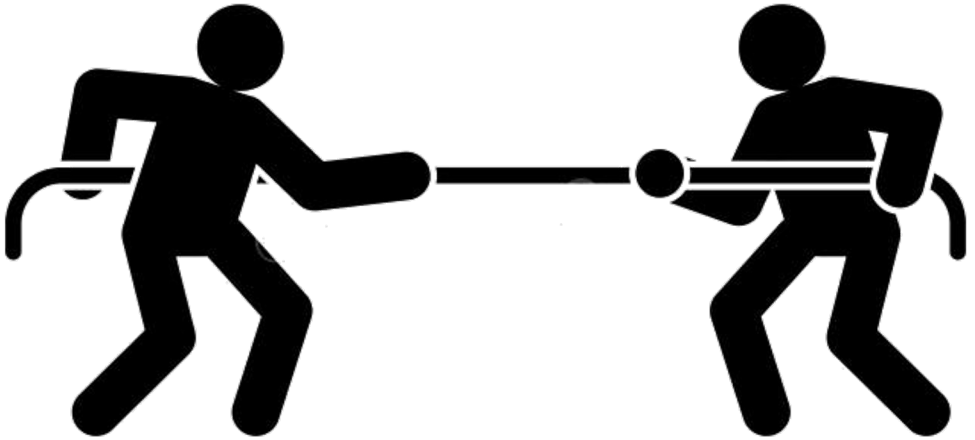
Optimization

Simulation

Types of Analysis

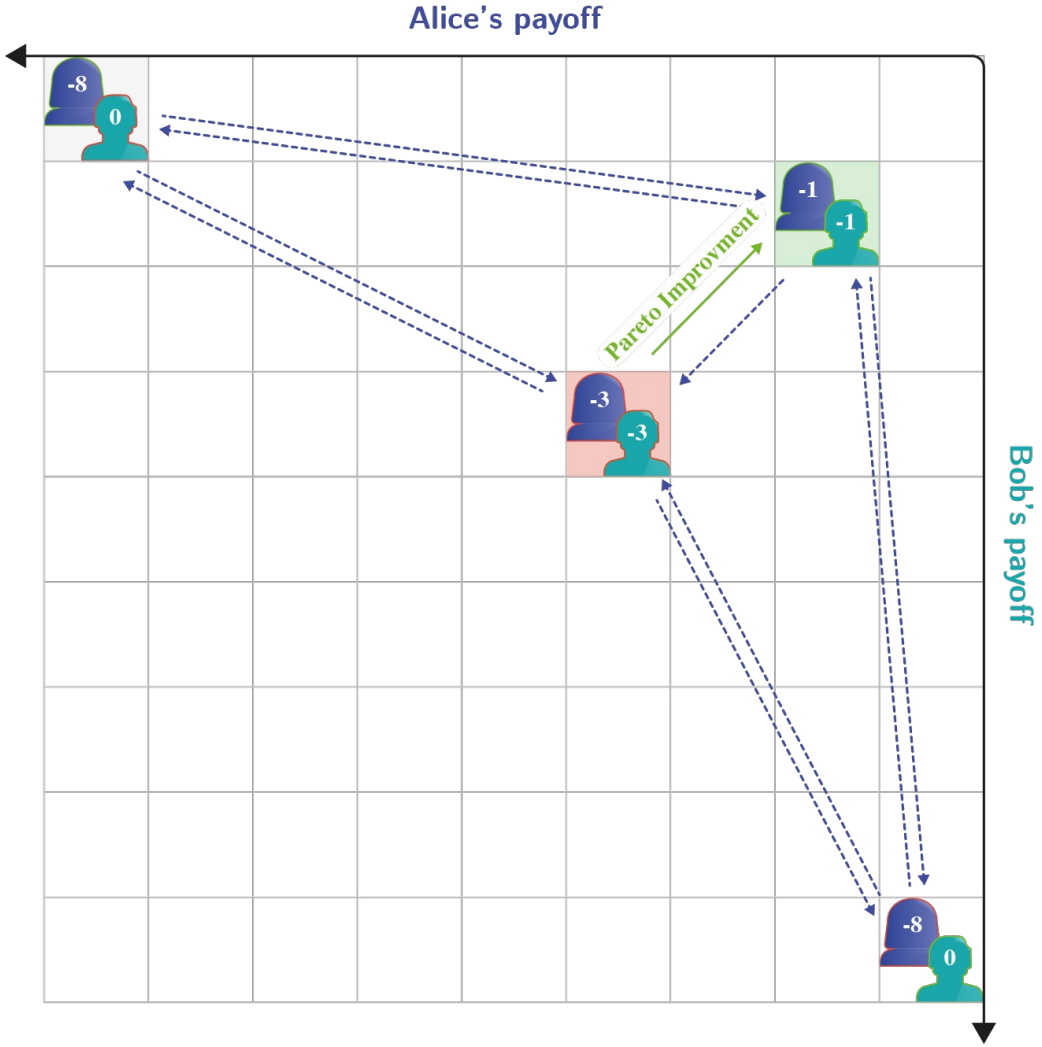
More than just yourself?

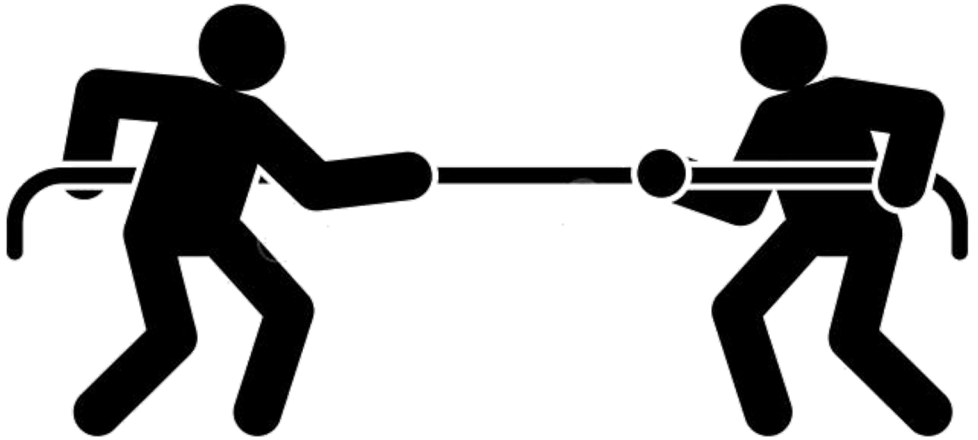
Corporative



Competitive

Two-person situation – Zero-Sum Game

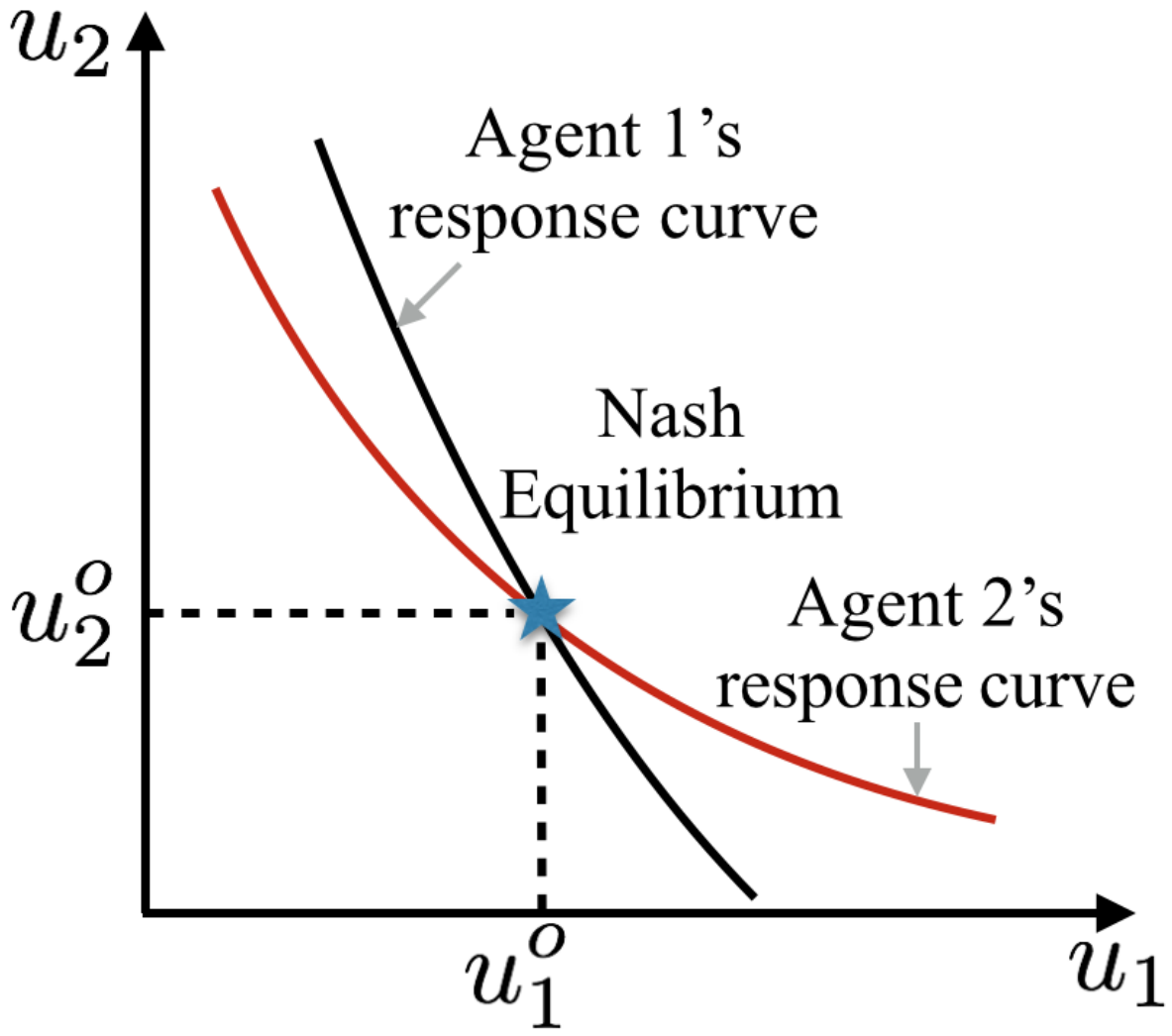




Competitive

https://www.aisafetybook.com/textbook/game-theory

Nash Equilibrium

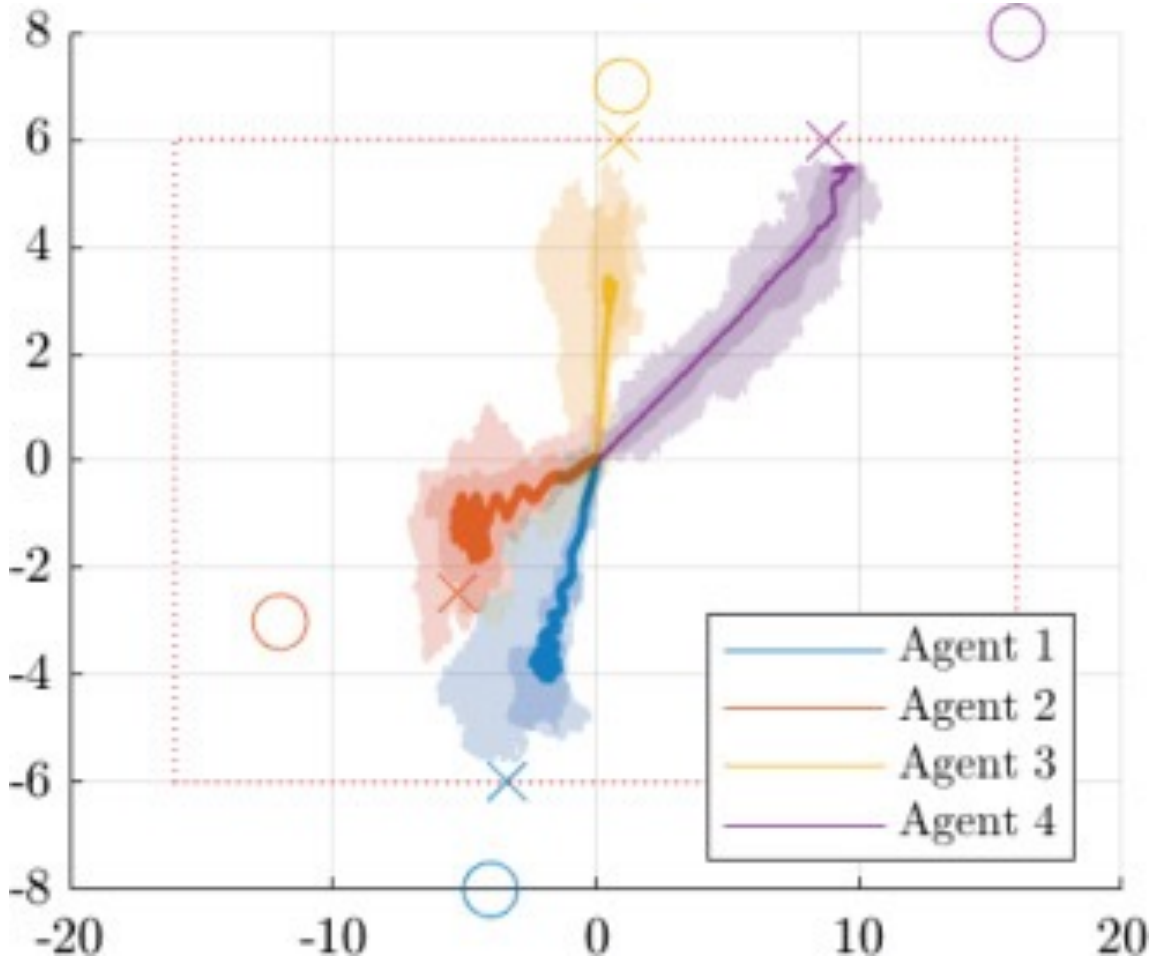


No player can improve their outcome by changing their strategy

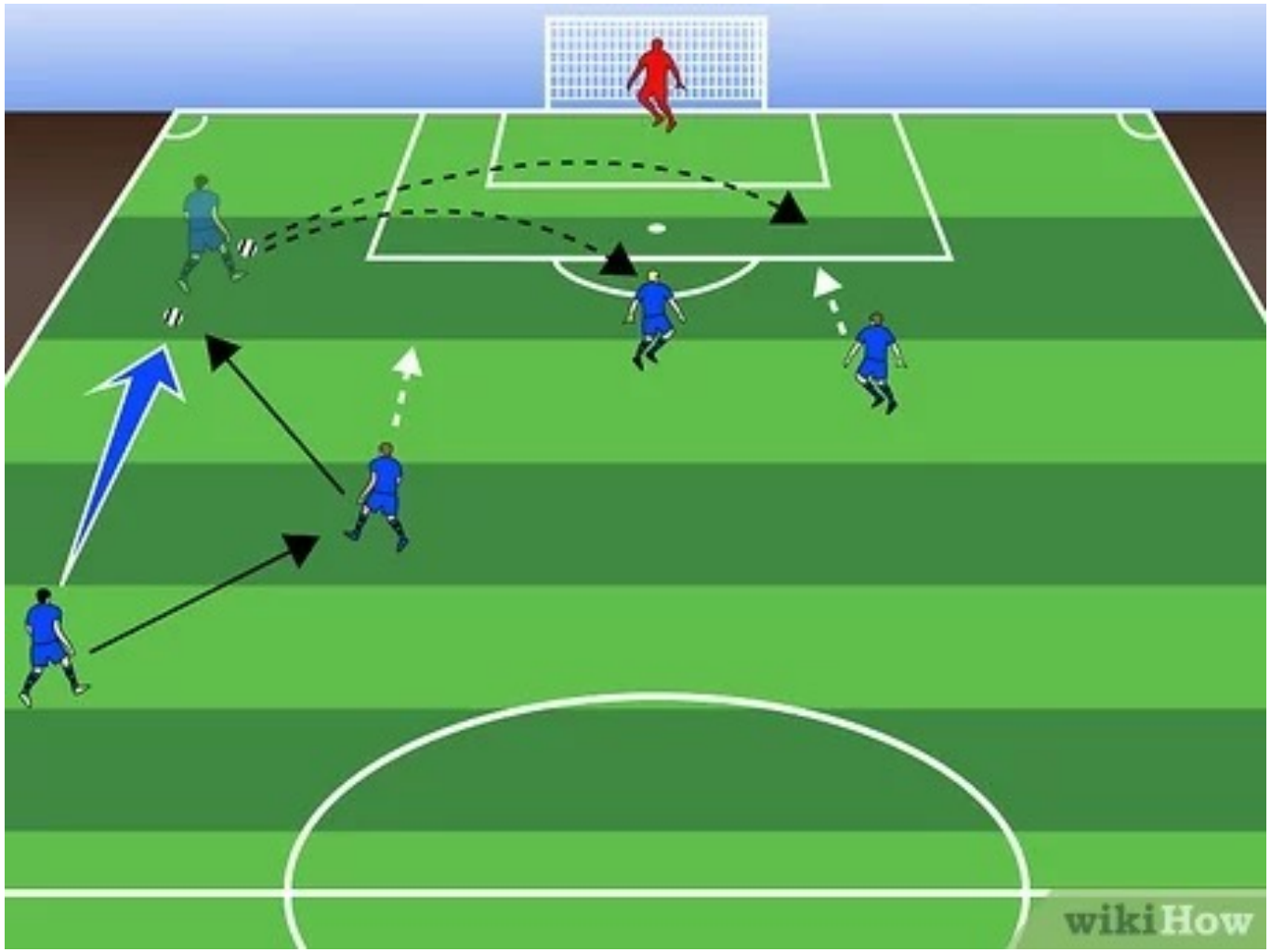
-- the solution of N-player noncooperative games.

-- hard to compute that *exact* point

What will happen if there are more than 2 people?

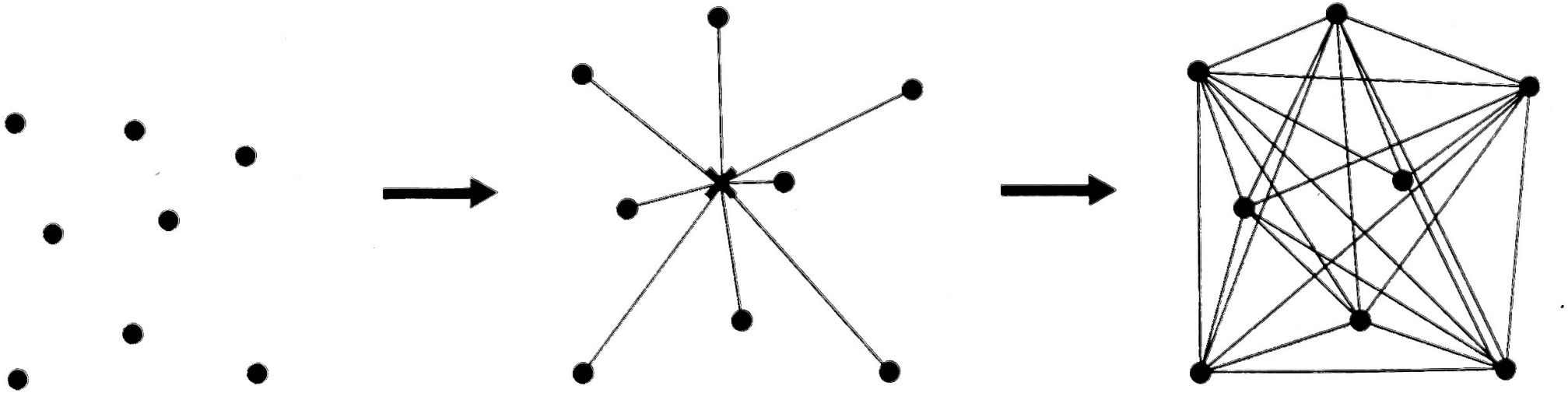


What will happen if the players needs collaborate?



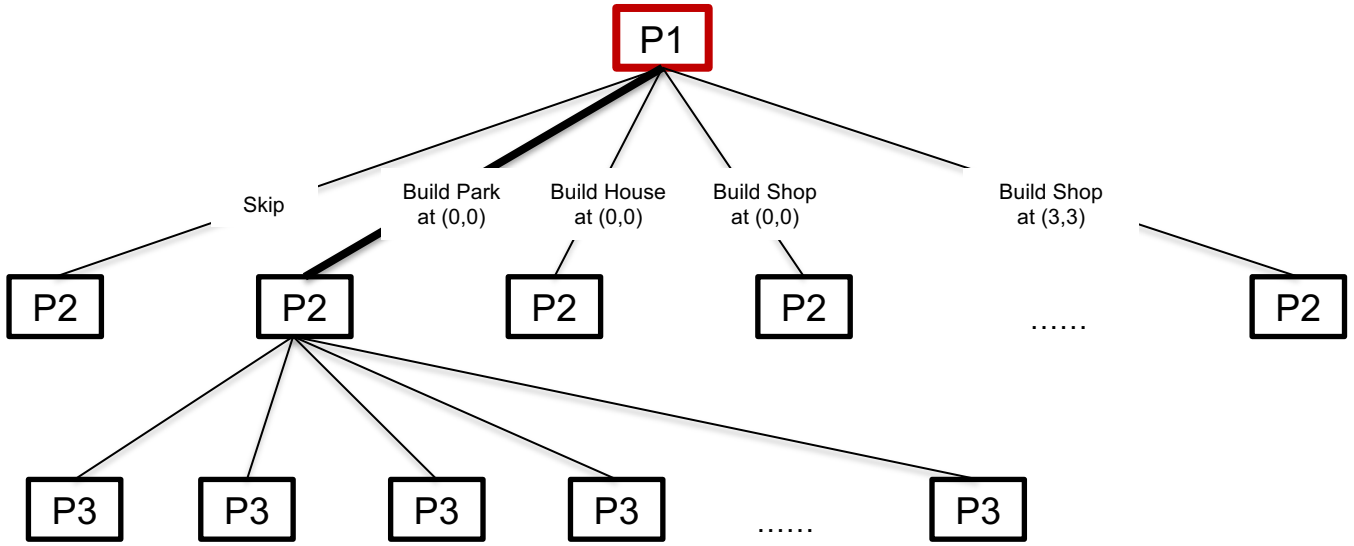
What will happen if the players have hierarchies?

The major ways human societies have been organized throughout history reveal a remarkably simple pattern that foreshadows how businesses are now changing.



Malone, Thomas W. *The future of work*. Audio-Tech Business Book Summaries, Incorporated, 2004

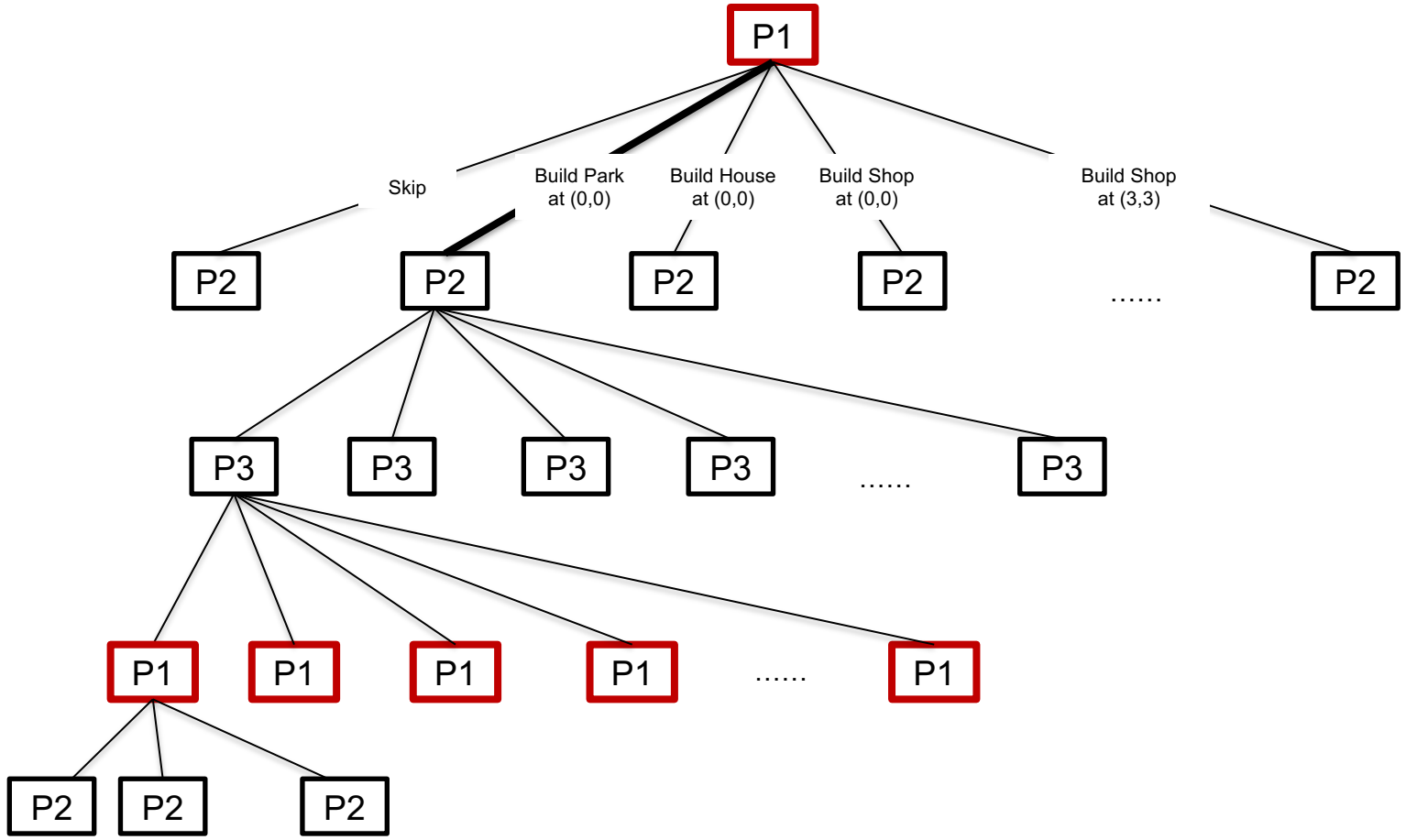
Intuition of making a series of decisions From Player 1's Perspective



Tree-form Decision Process (TFDP)

Intuition of making a series of decisions

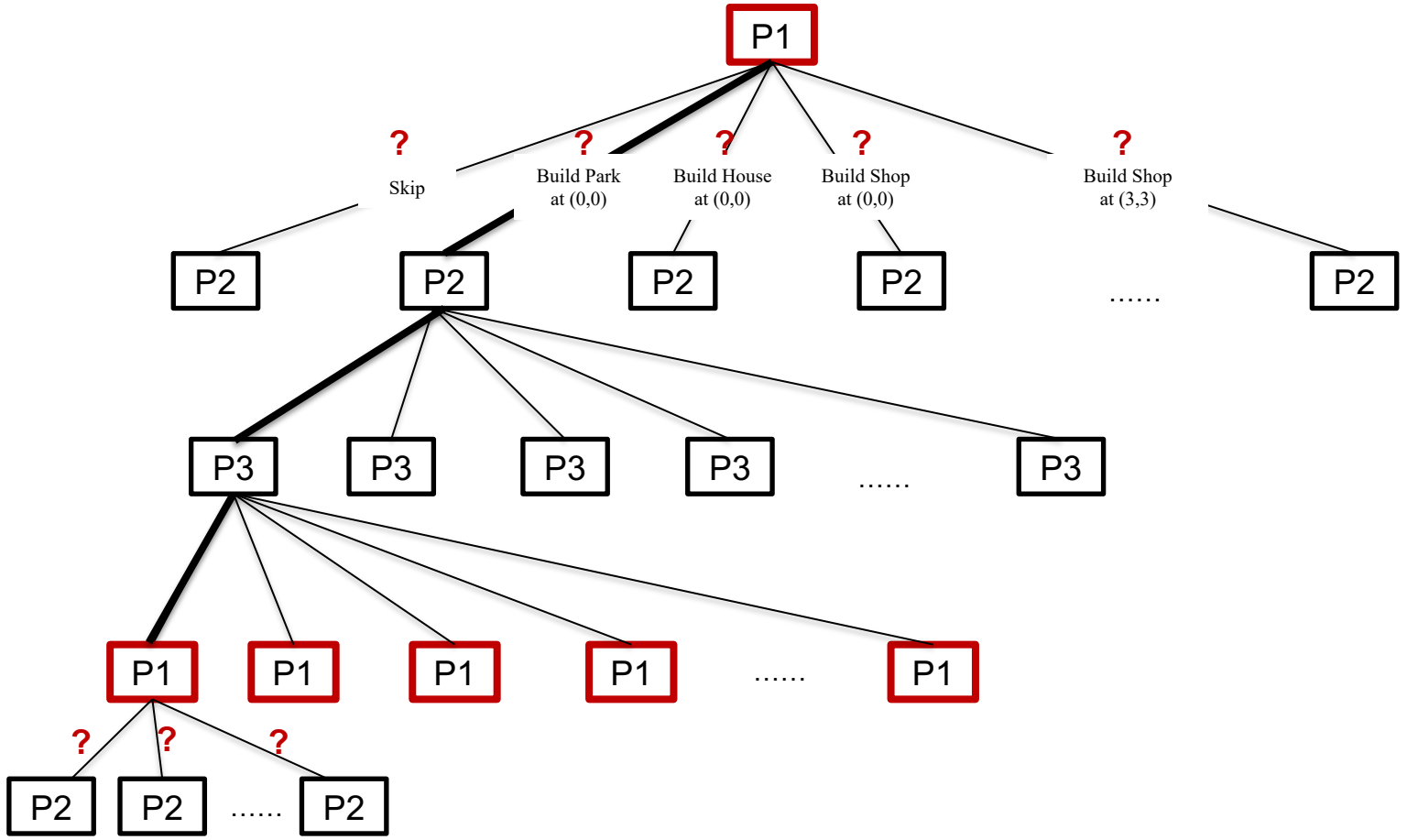
From Player 1's Perspective



Tree-form Decision Process (TFDP)

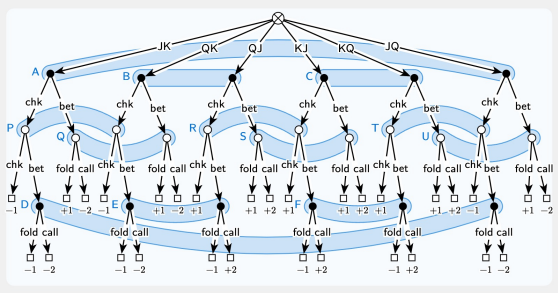
Intuition of making a series of decisions

From Player 1's Perspective



Tree-form Decision Process (TFDP)

Plotting Decisions with Hidden Information: Game Tree vs Tree Form Decision Process



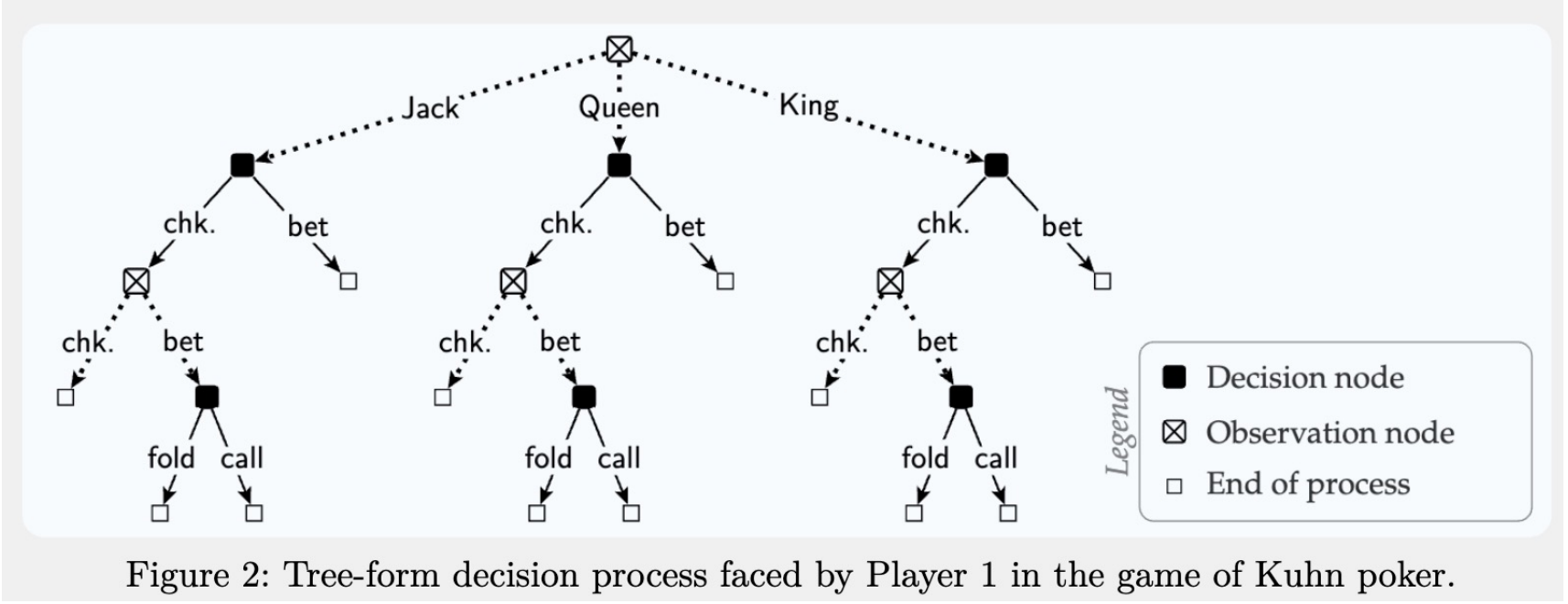
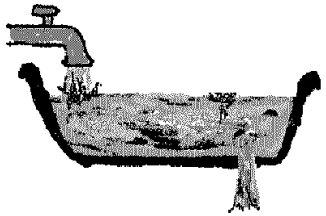
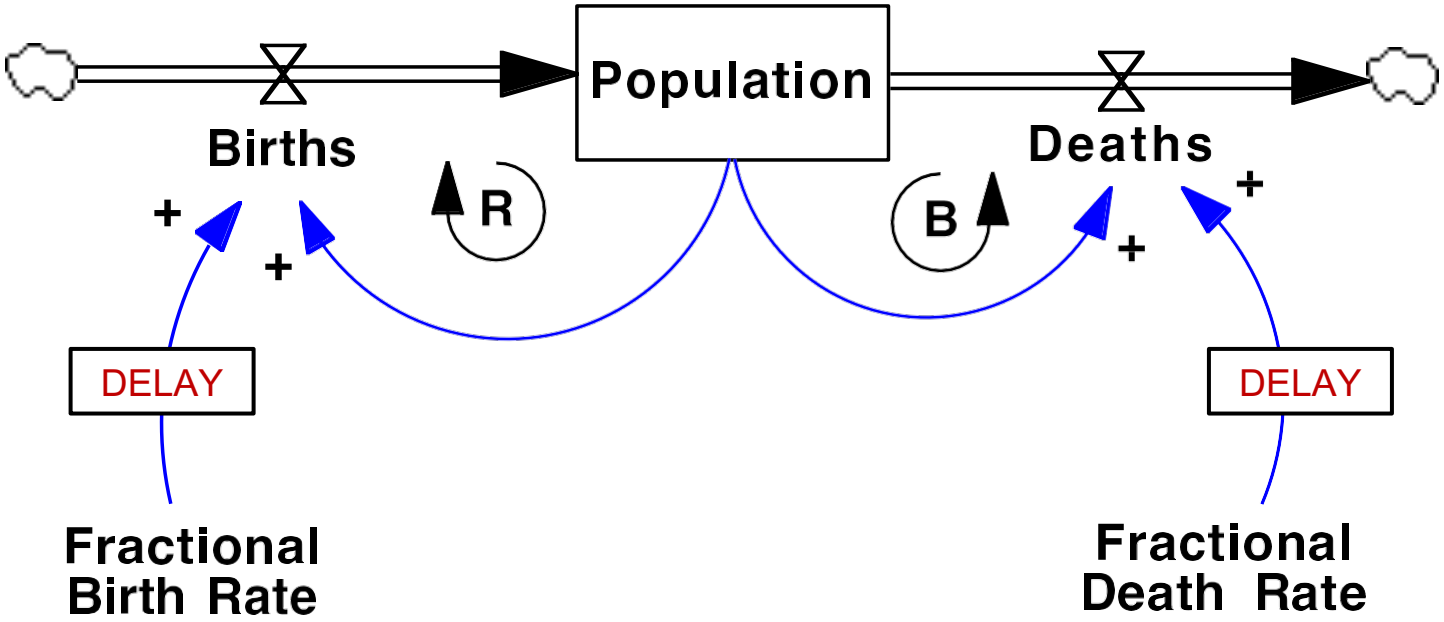


Figure 2: Tree-form decision process faced by Player 1 in the game of Kuhn poker.

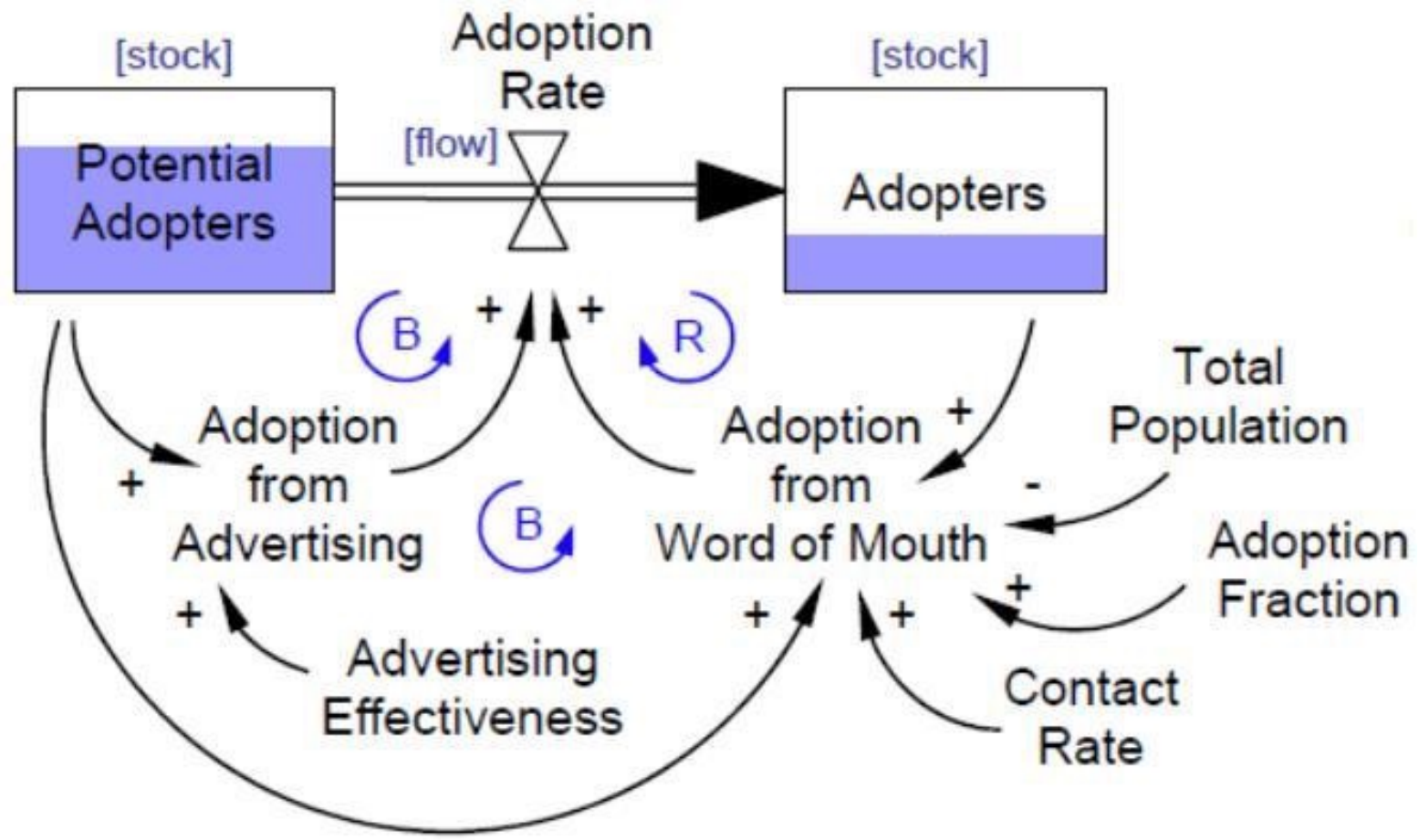
Decision Tree

Basic Elements: Delay



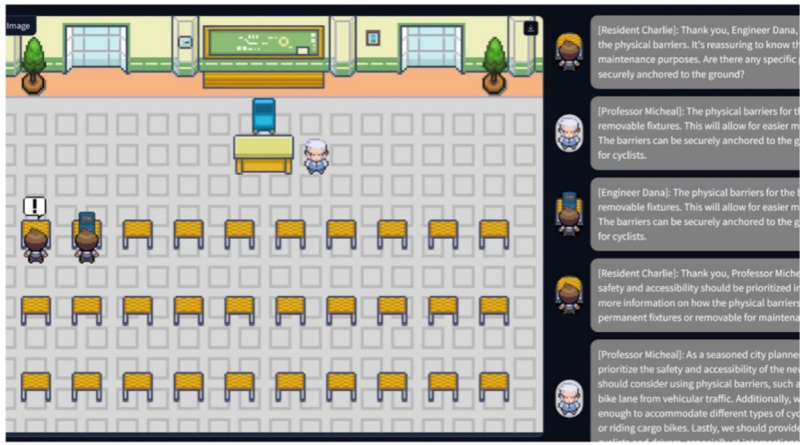


Classic Models



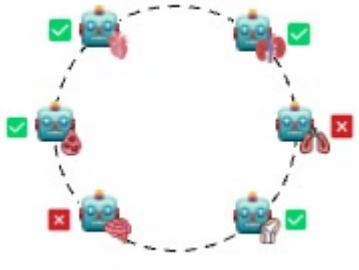
Bass Diffusion Model

Multi-Generative Agent System

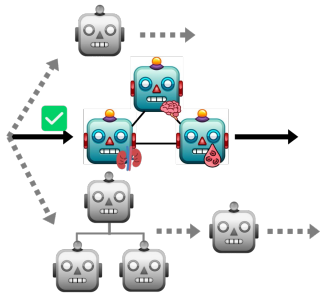


Multi-Generative Agent System proposed in recent researches [Park et al., 2023] (left) and [Chen et al., 2023] (right)

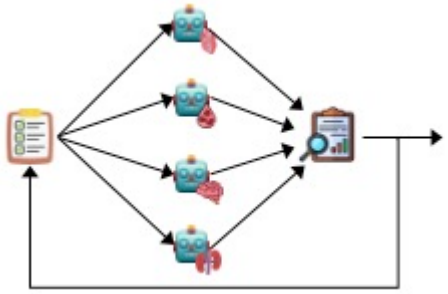
Agents-level Institution: Multi-Agent Debate Frameworks



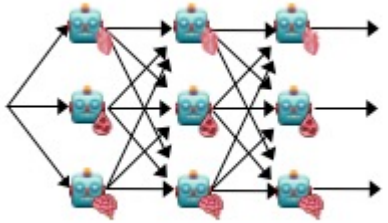
Roundtable Voting



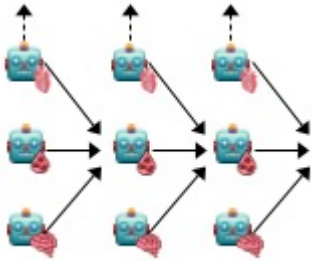
Branching-Verify



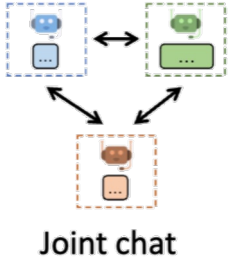
Diverge-Conclude



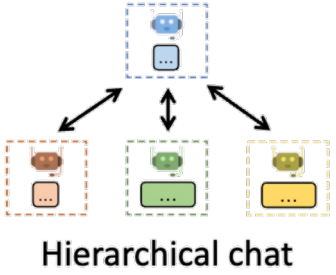
Exhaustive Inquiry



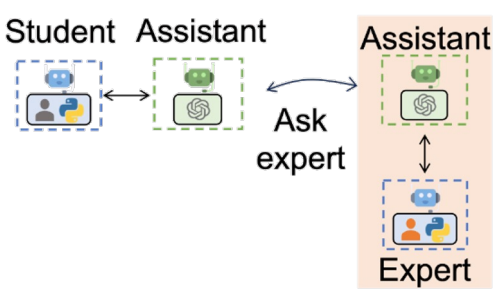
Induction



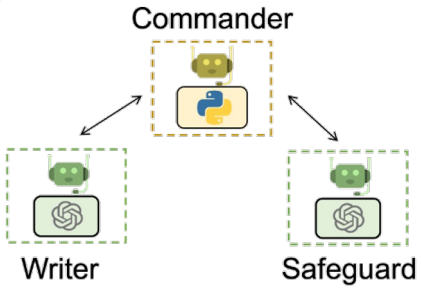
Joint Chat



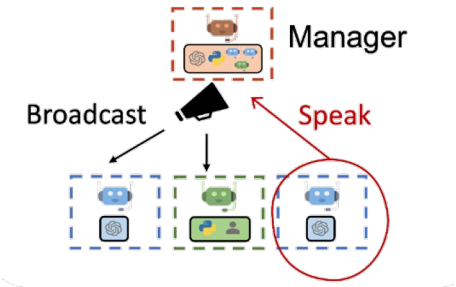
Hierarchical Chat



Expert Consulting



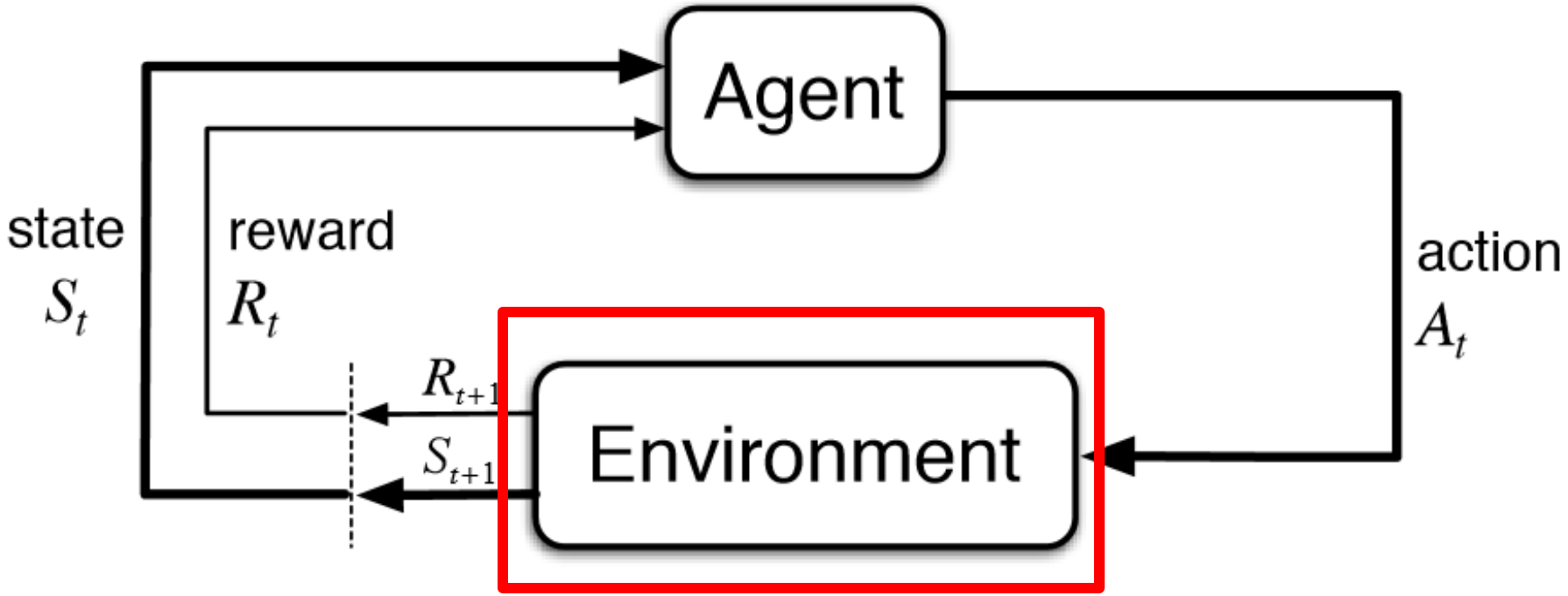
Creator–Executer-Criticizer



Broadcasting

https://arxiv.org/abs/2404.15155
https://arxiv.org/abs/2308.08155

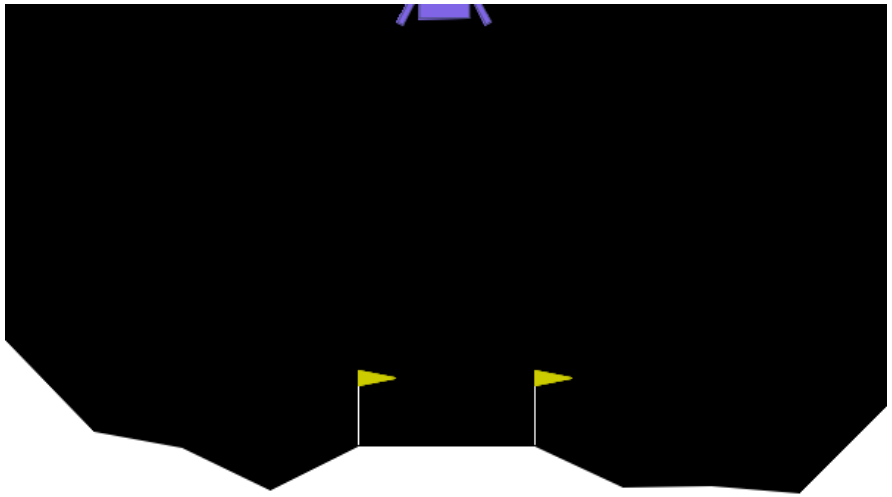
Where does agents learn?



OpenAI Gymnasium

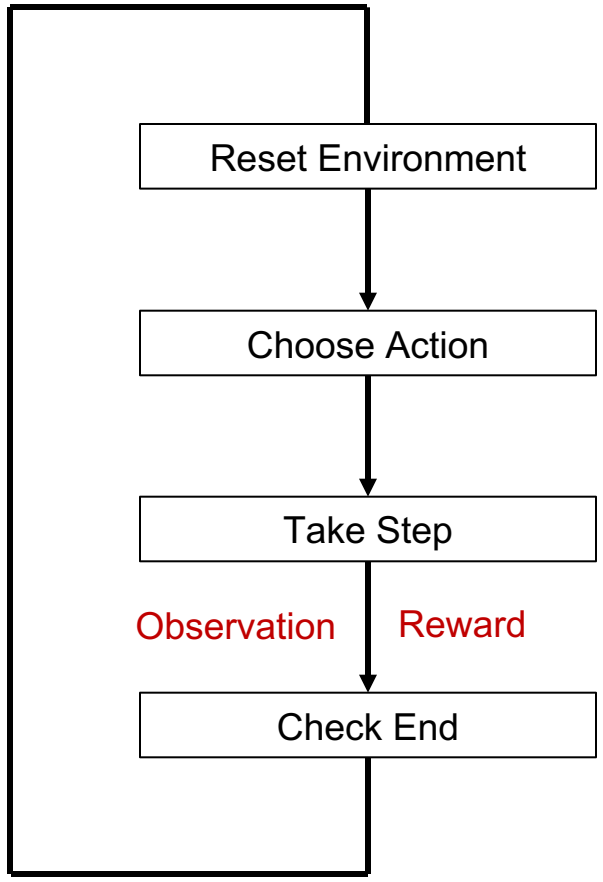
Gymnasium

An API standard for reinforcement learning with a diverse collection of reference environments



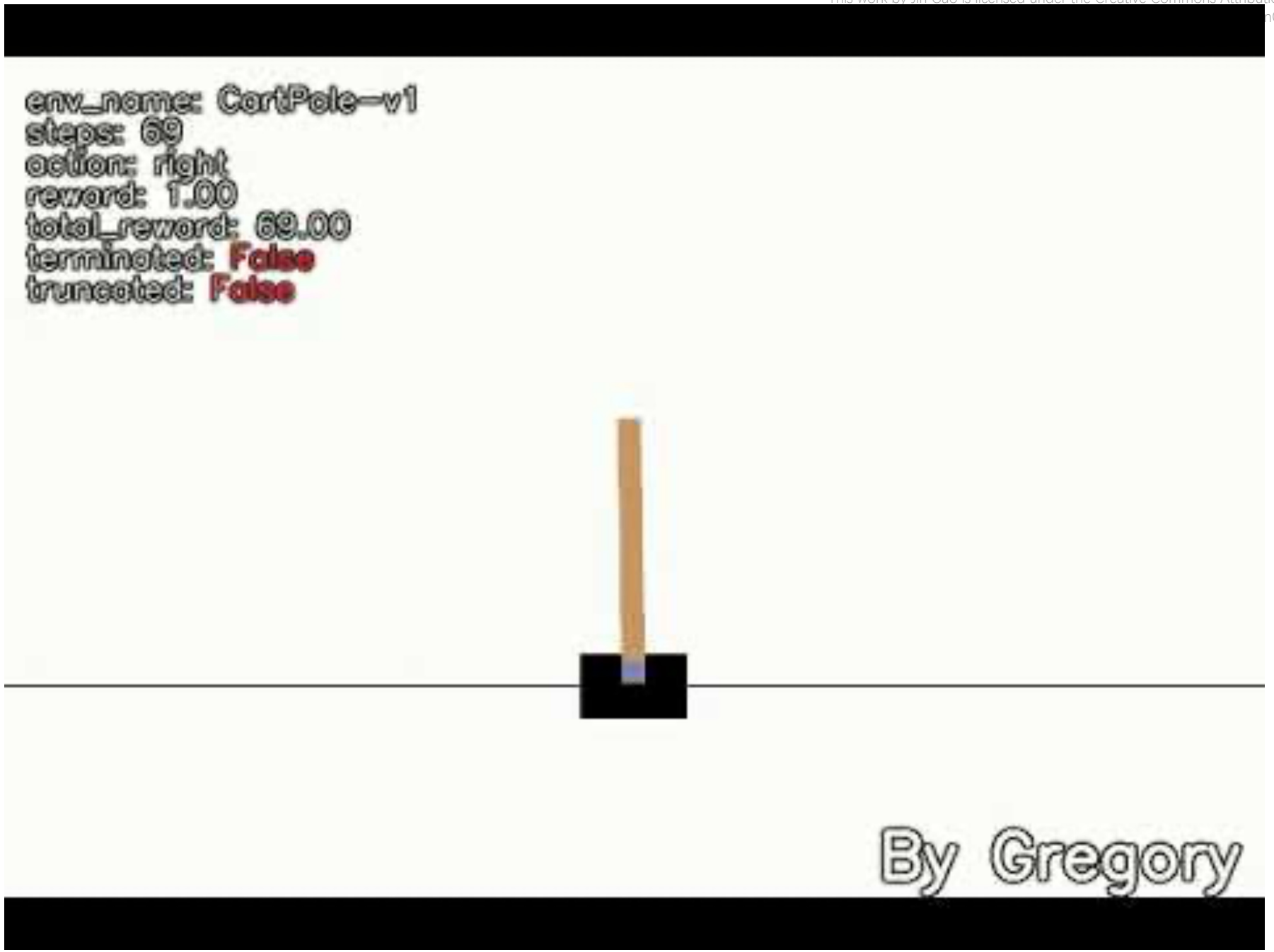
https://gymnasium.farama.org/index.html

Execution Loop

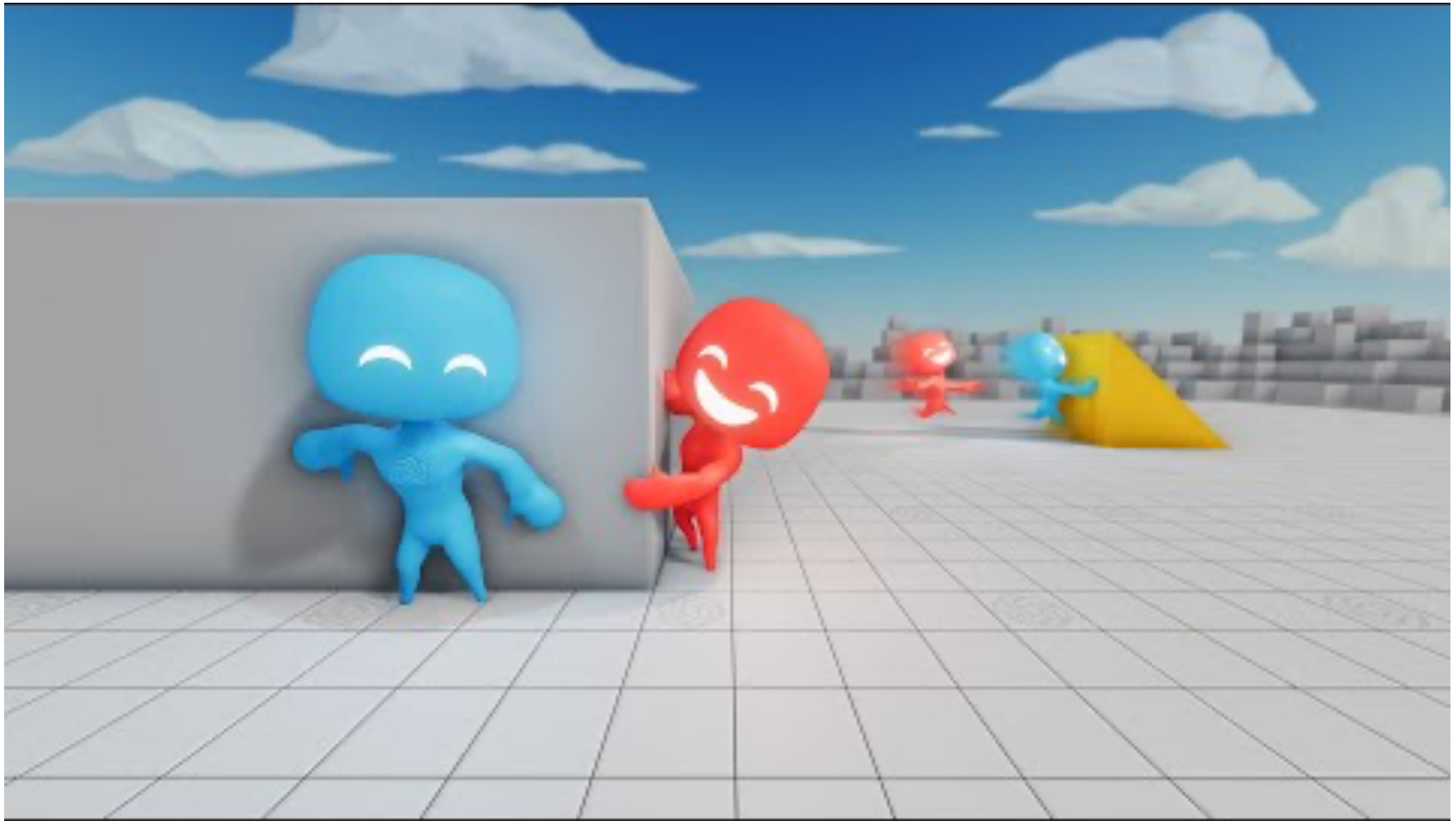



```python
import gymnasium as gym

# Initialise the environment
env = gym.make("LunarLander-v3", render_mode="human")

# Reset the environment to generate the first observation
observation, info = env.reset(seed=42)
for _ in range(1000):
    # this is where you would insert your policy
    action = env.action_space.sample()

    # step (transition) through the environment with the action
    # receiving the next observation, reward and if the episode has terminated or truncated
    observation, reward, terminated, truncated, info = env.step(action)

    # If the episode has ended then we can reset to start a new episode
    if terminated or truncated:
        observation, info = env.reset()

env.close()
```

https://gymnasium.farama.org/index.html

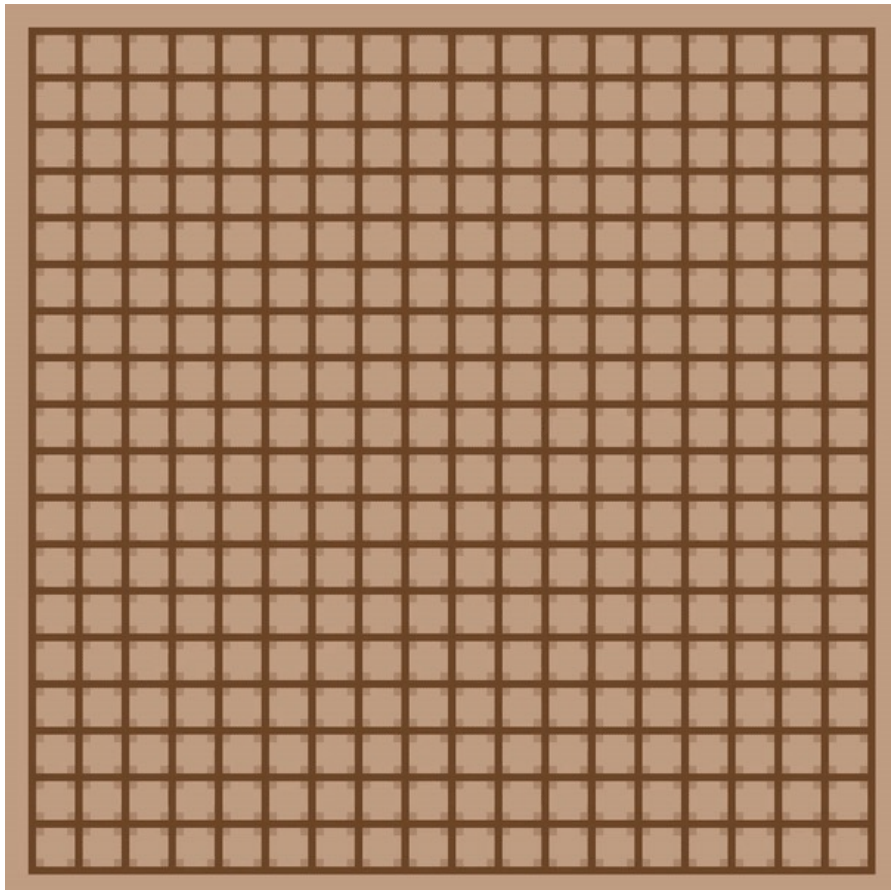


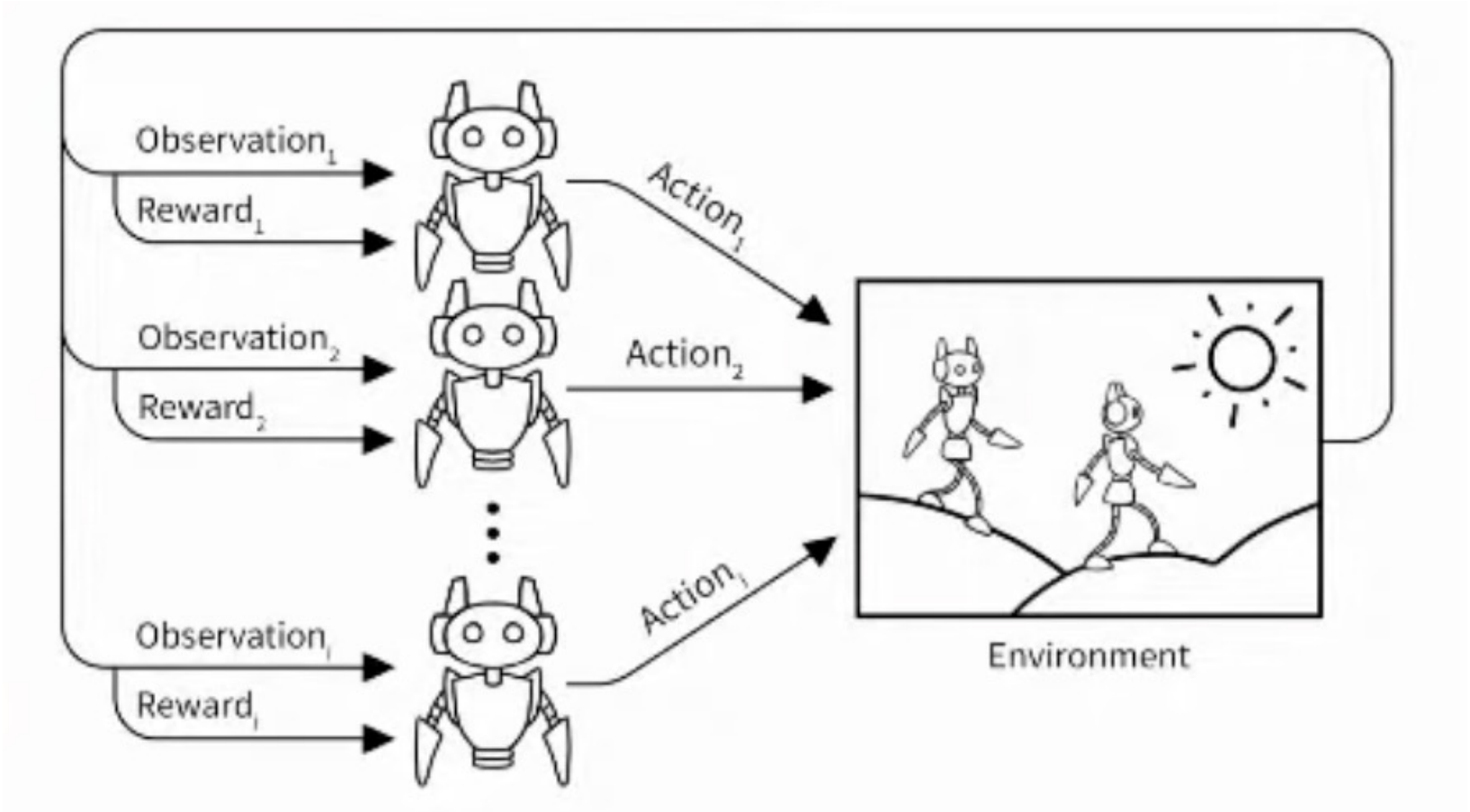
What about multi-agent?



PettingZoo

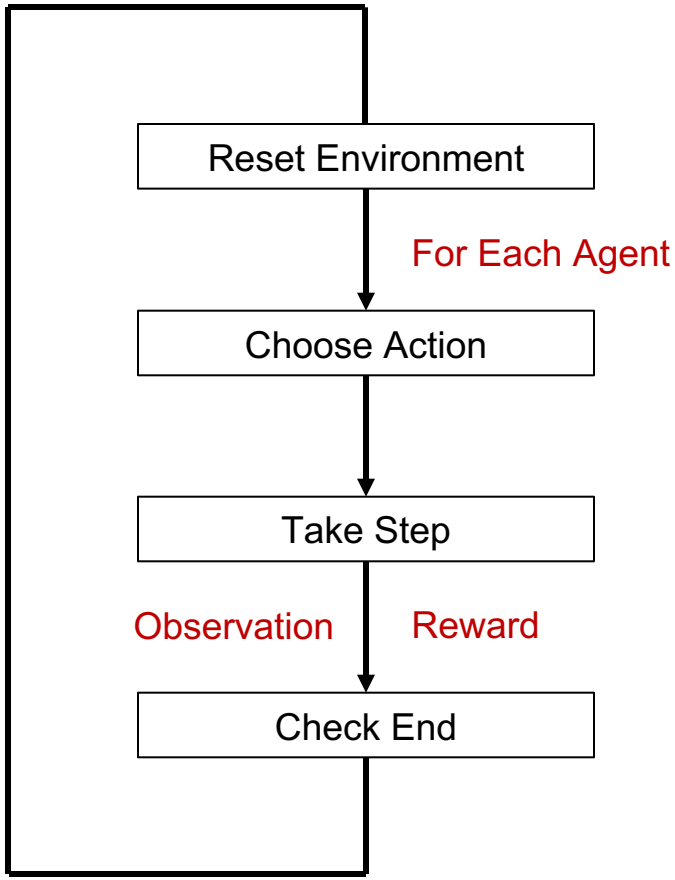
An API standard for multi-agent reinforcement learning.





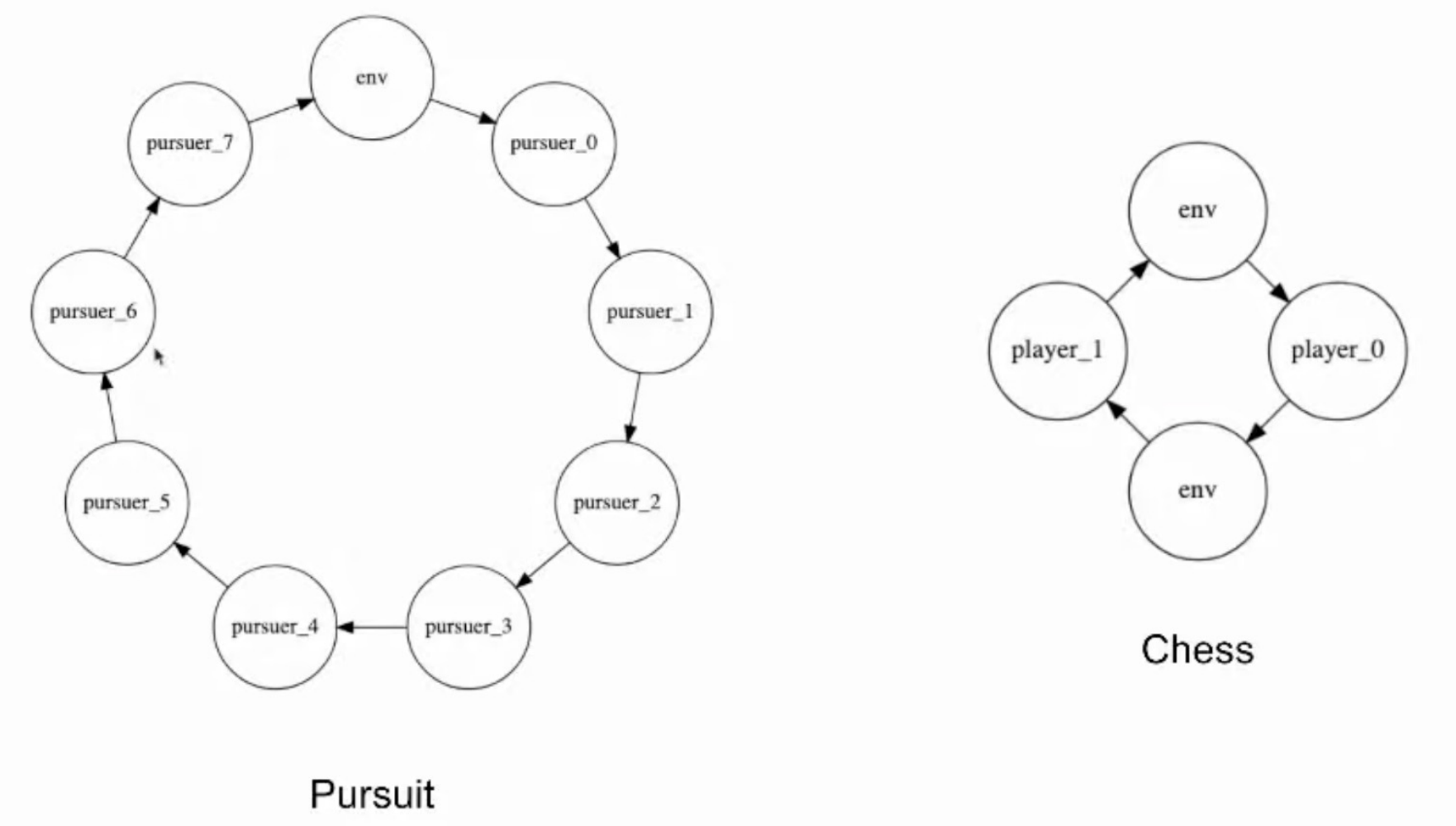
https://www.youtube.com/watch?v=IMpf_X1IN_0&t=1584s

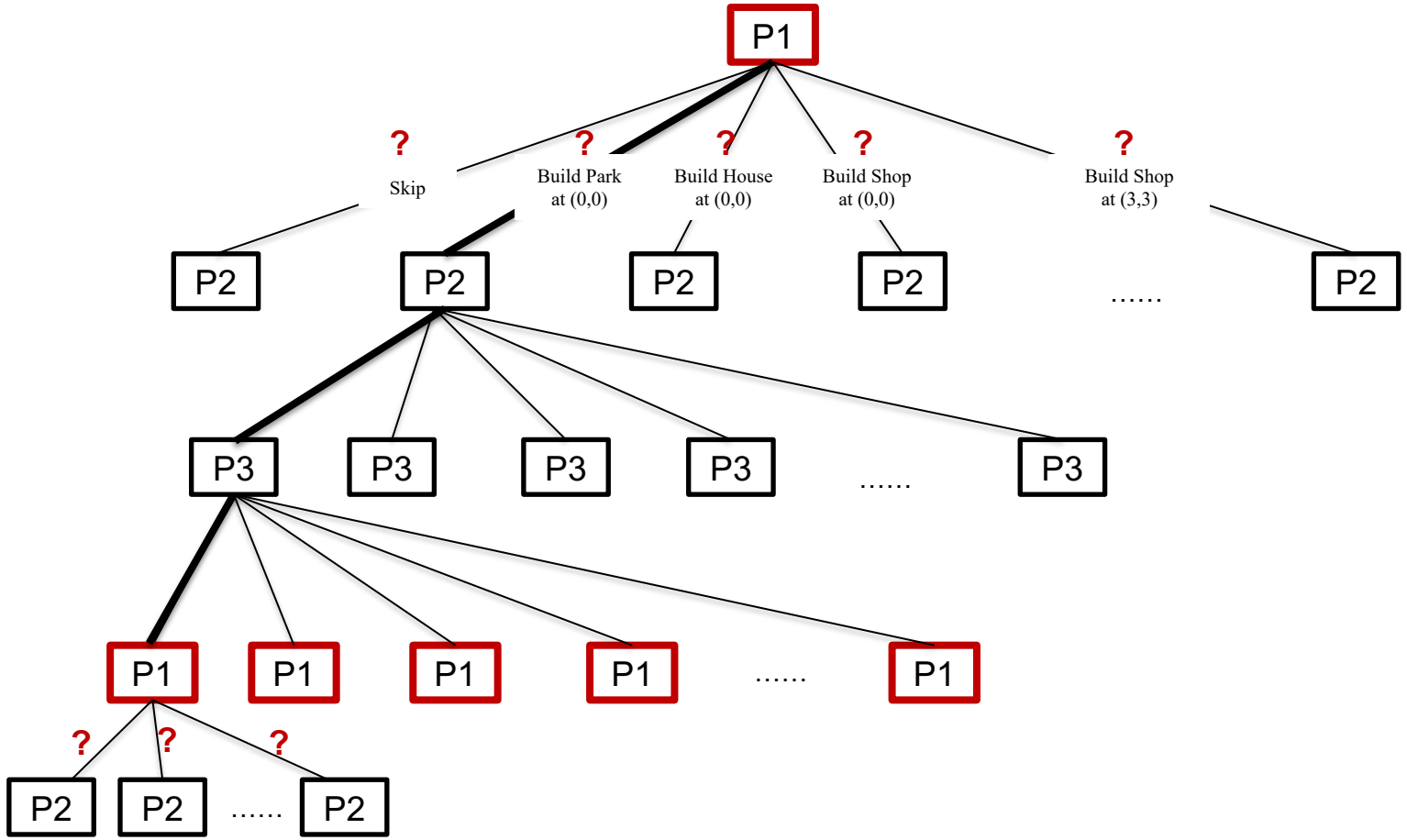
Execution Loop




```python
from pettingzoo.butterfly import knights_archers_zombies_v10
env = knights_archers_zombies_v10.env(render_mode="human")
env.reset(seed=42)
for agent in env.agent_iter():
    observation, reward, termination, truncation, info = env.last()
    action = policy(observation, agent)
    env.step(action)
```

Agent-Environment Cycle

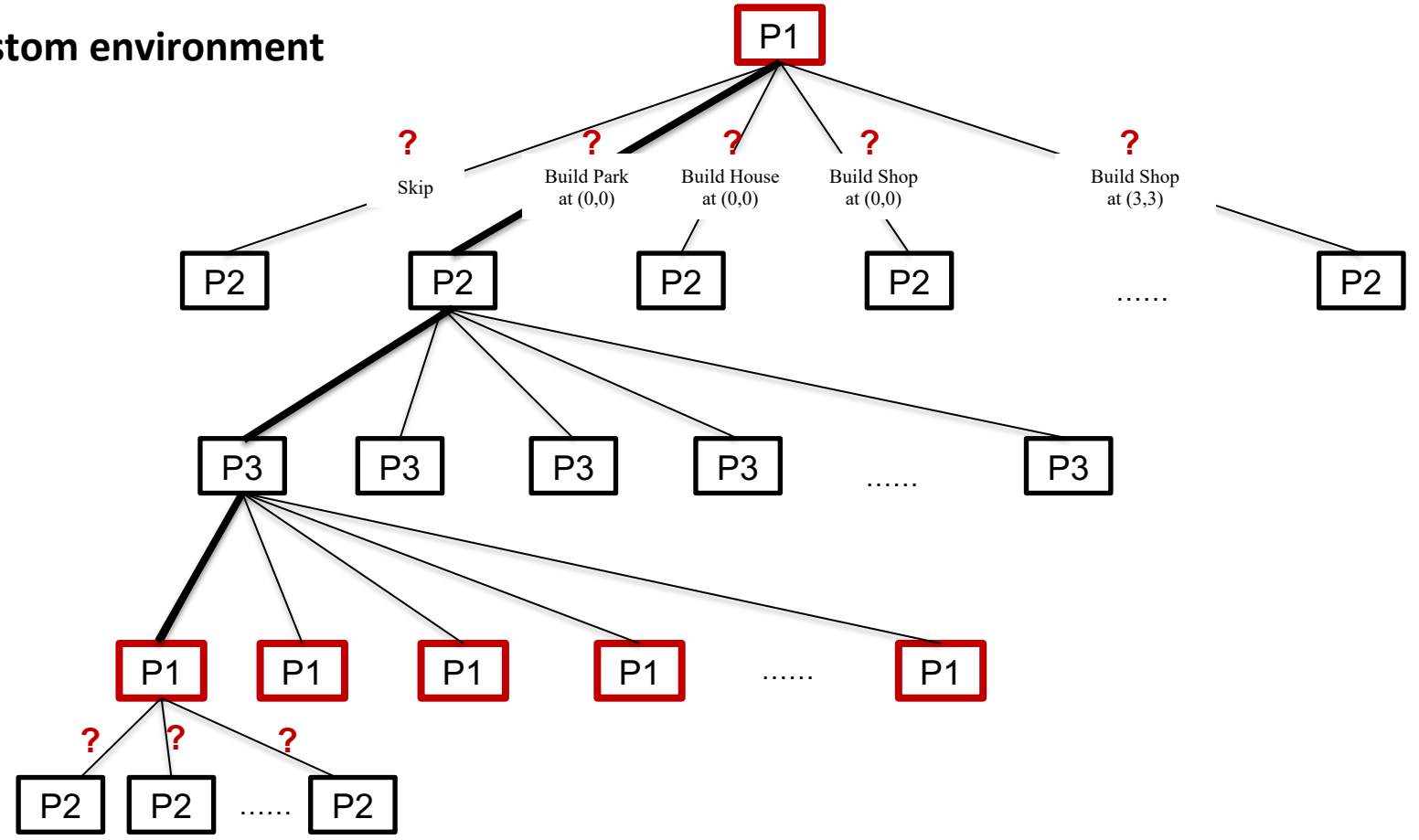




Tree-form Decision Process (TFDP)

Training in the environment

Using EPyMARL Library with custom environment



https://github.com/uoe-agents/epymarl

Training Outcome? The best game strategies

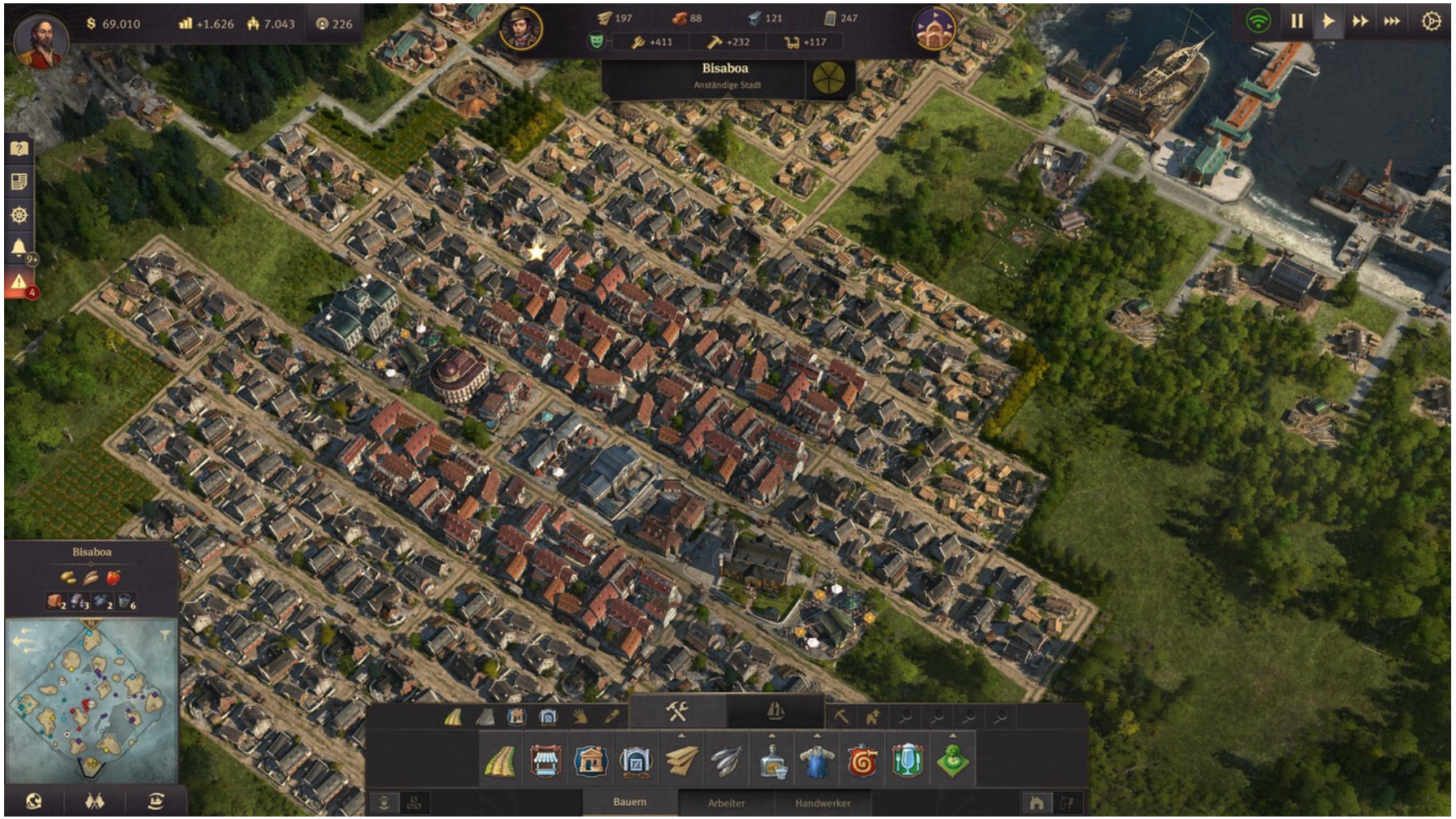
Cities Skylines

Training Outcome? The best game strategies

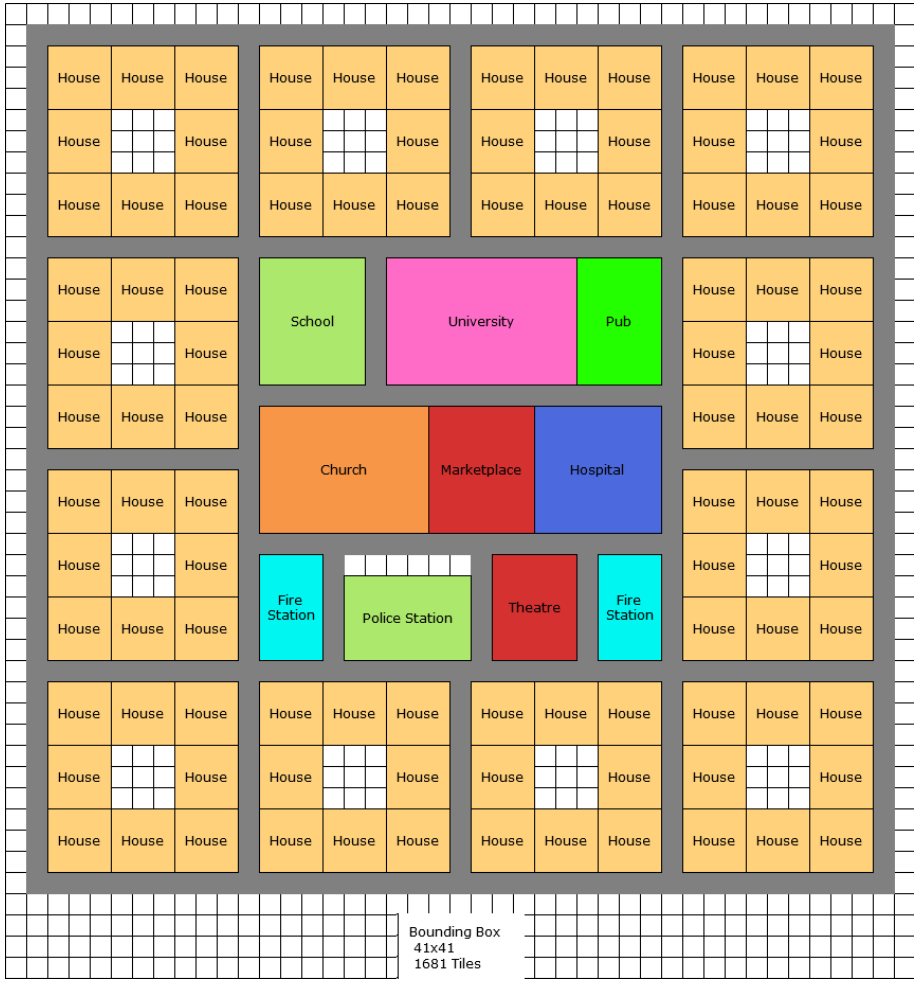
Cities Skylines

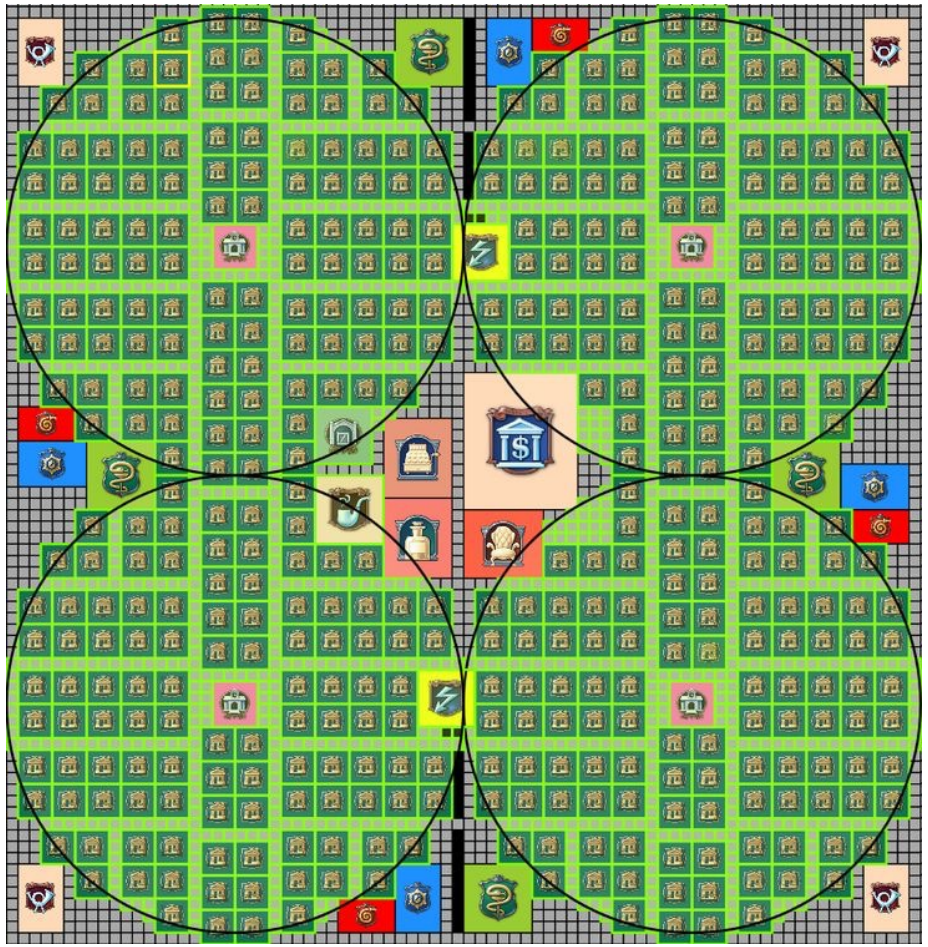
Training Outcome? The best game strategies

ANNO 1800

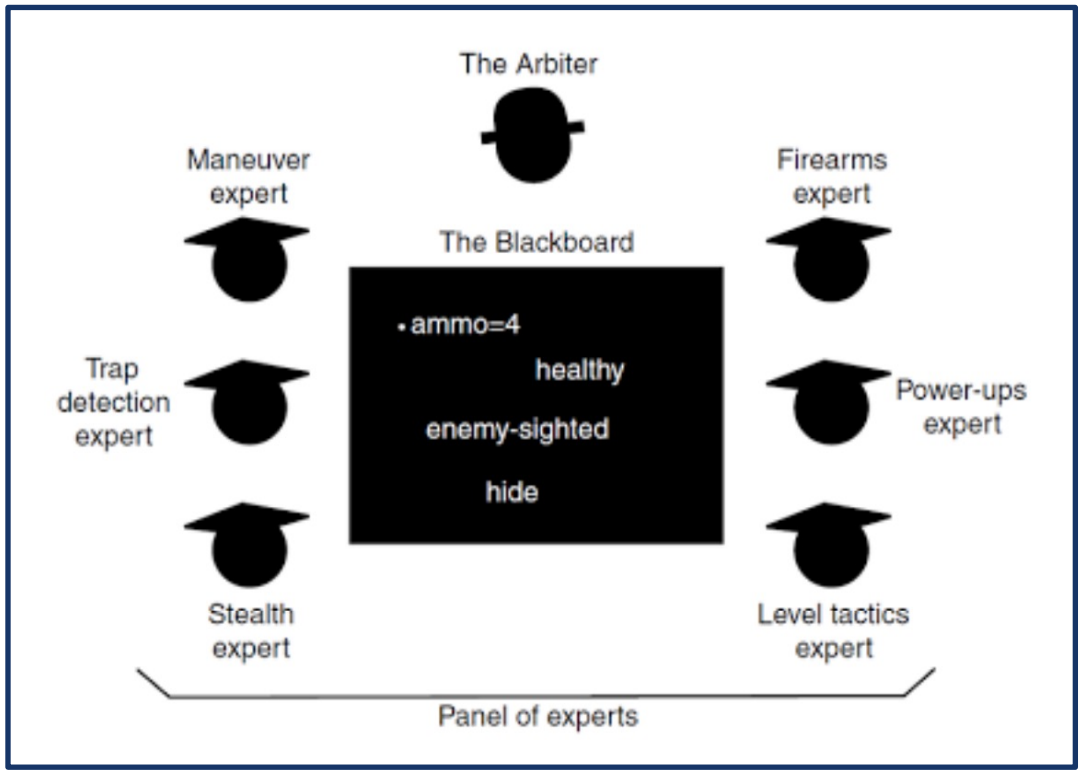


Training Outcome? The best game strategies

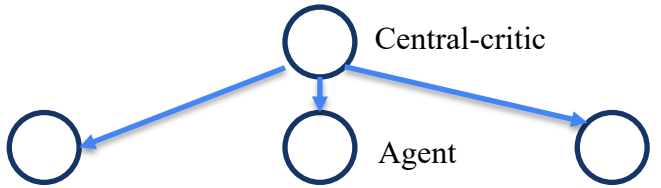
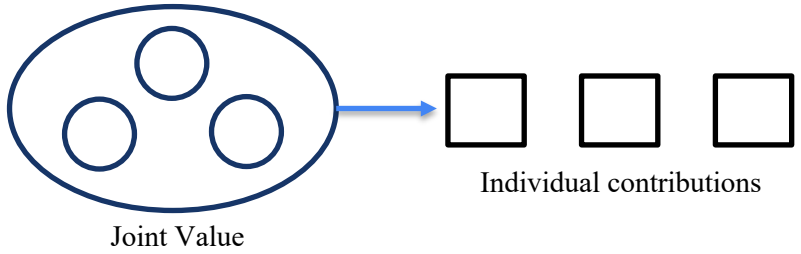
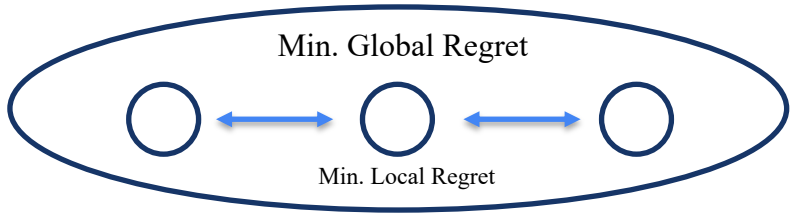




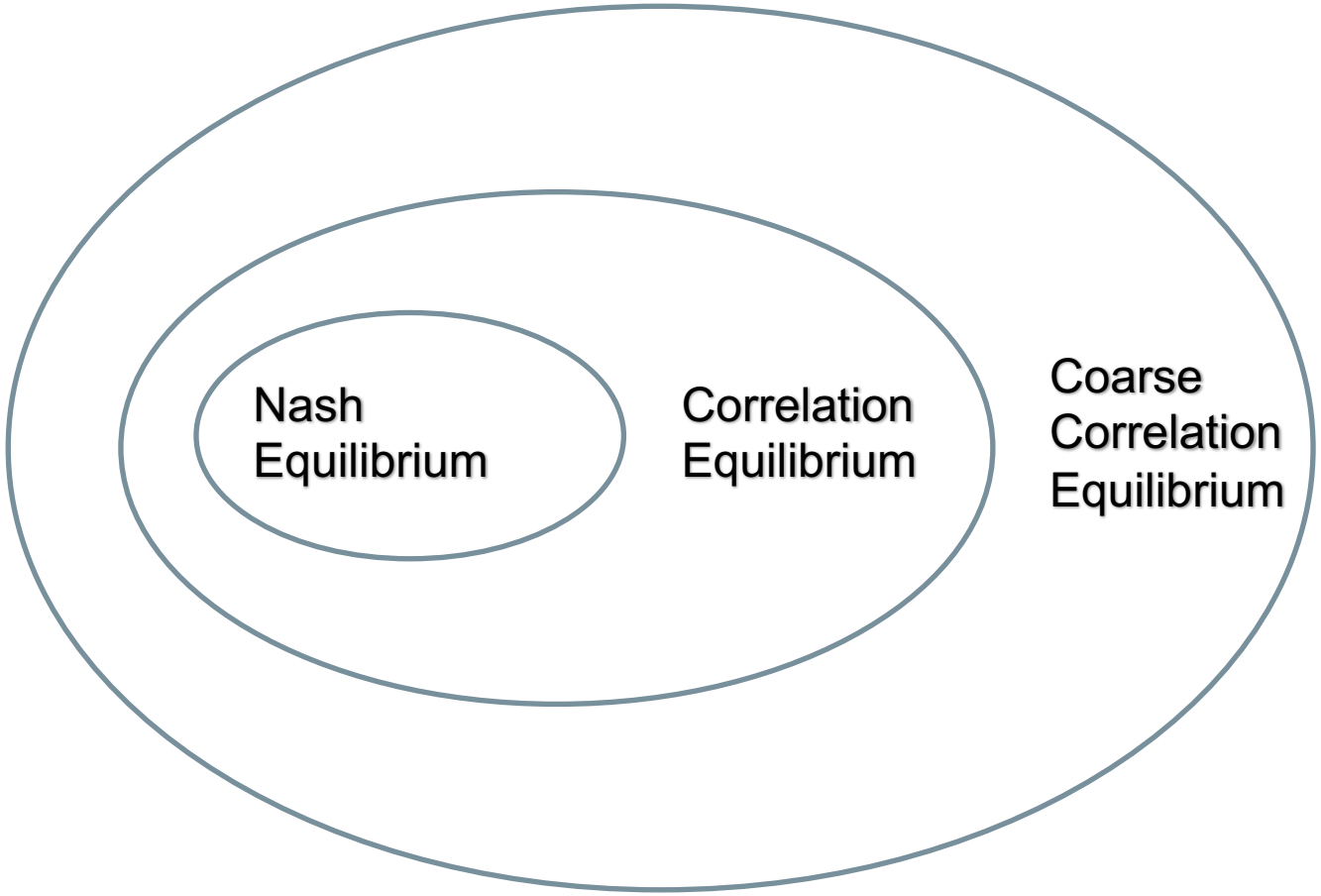
Collaboration between multiple players - Blackboard Model



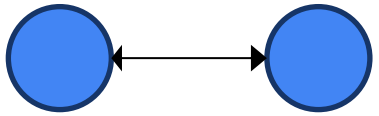
Algorithms Inspired by Social Structures

Category	Description	Illustrations	Algorithms
Independent Learning	Each agent learns independently, treating others as part of the environment.		IQL, IPG, IA2C, ITRPO, IPPO
Centralized Critic	Uses a centralized critic to estimate value functions while policies remain decentralized for execution.		COMA, MATRPO, MAPPO, MADDPG, MAA2C
Value Decomposition	Decomposes joint value functions into individual contributions to facilitate coordination among agents.		QMIX, VDN, QTRAN
Regret-Based Methods	Minimizing regret over iterations, commonly used in solving extensive-form games and multi-agent strategies.		CFR, MC-CFR, MWU, Regret Matching, Regret Matching+

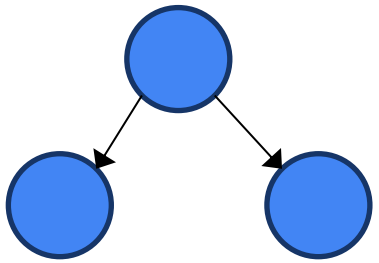
Existence of a good strategy : Game Theory 2.0



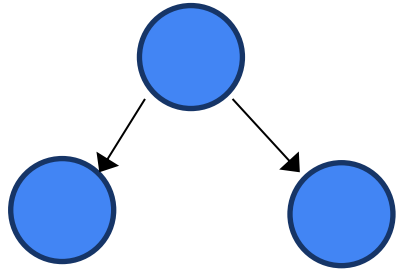
NE:



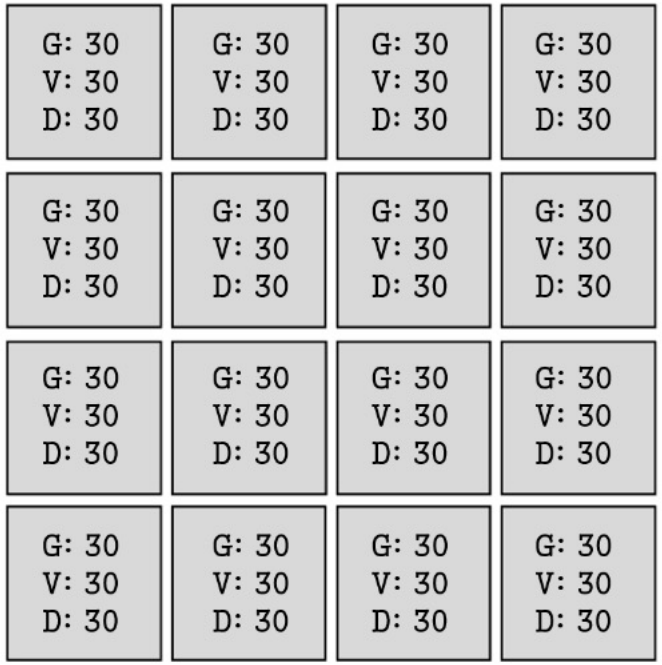
CE:



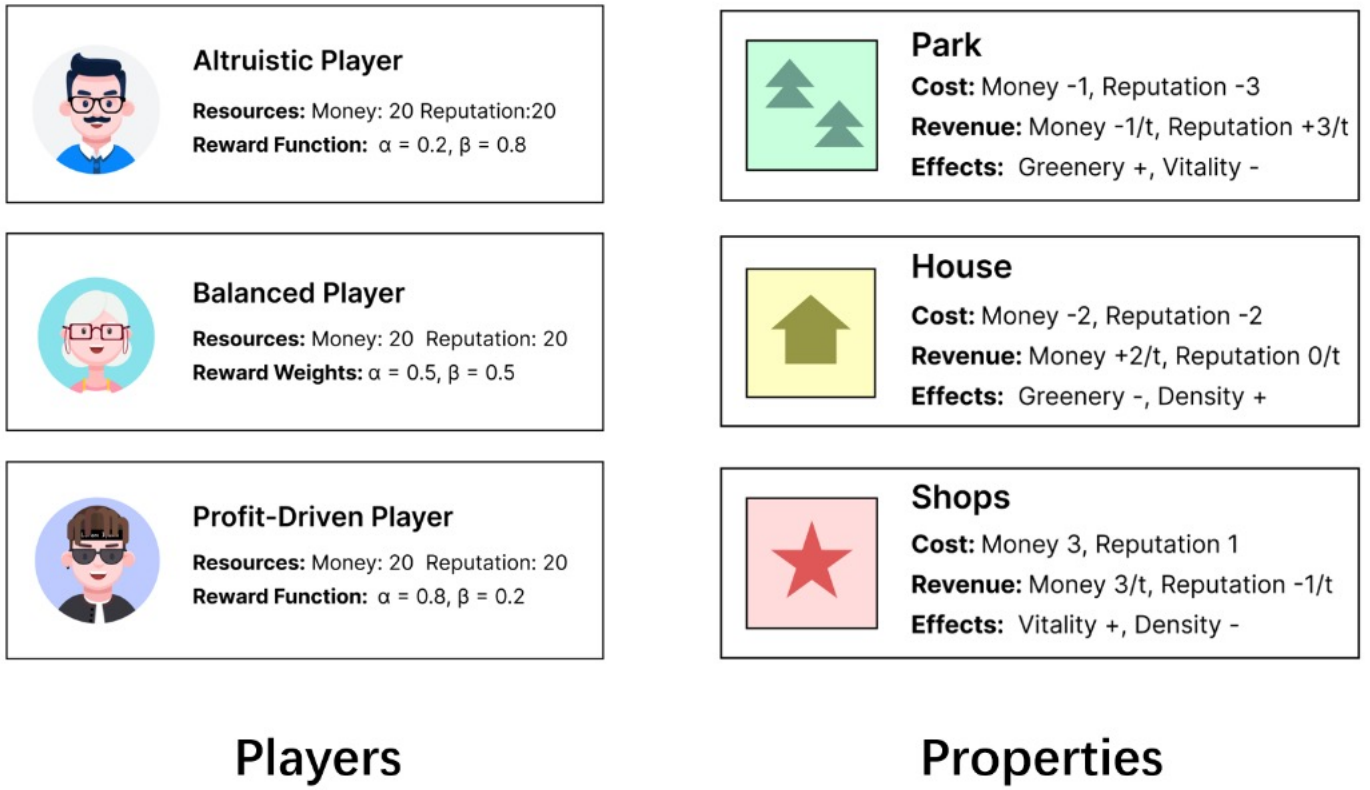
CCE:



Easier to compute (converge) !

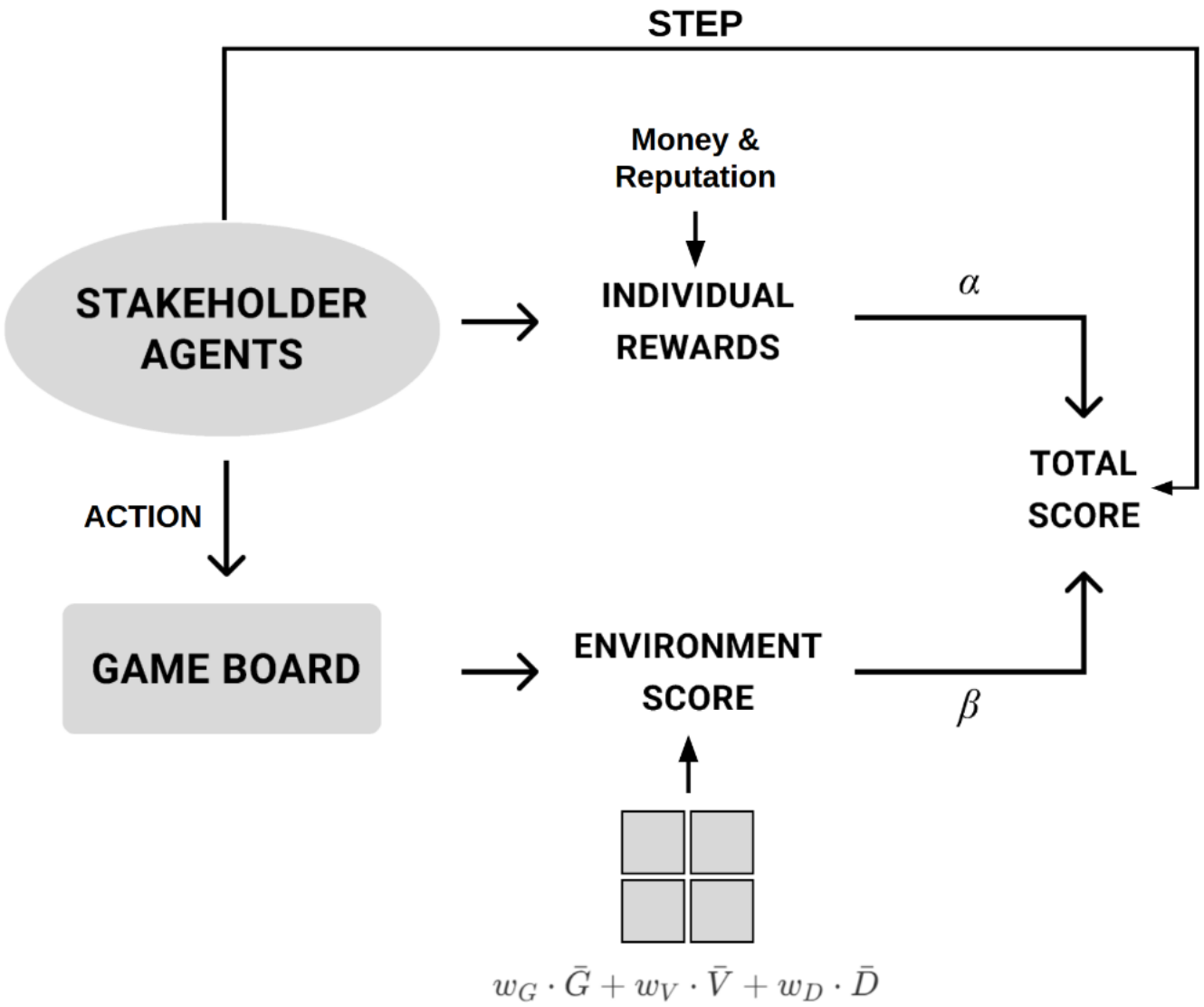


Initial Game Board



Players and Properties

Case Study: Minimalist Simcity



Environment Parameters

Avg. Greenery : 30
Avg. Vitality: 30
Avg. Density : 30

Env Score:

30

Personal: 0
Combined: 24

Personal: 0
Combined: 15

Personal: 0
Combined: 6

G: 30 V: 30 D: 30	G: 30 V: 30 D: 30	G: 30 V: 30 D: 30	G: 30 V: 30 D: 30
G: 30 V: 30 D: 30	G: 30 V: 30 D: 30	G: 30 V: 30 D: 30	G: 30 V: 30 D: 30
G: 30 V: 30 D: 30	G: 30 V: 30 D: 30	G: 30 V: 30 D: 30	G: 30 V: 30 D: 30
G: 30 V: 30 D: 30	G: 30 V: 30 D: 30	G: 30 V: 30 D: 30	G: 30 V: 30 D: 30

Environment Parameters

Avg. Greenery : 30
Avg. Vitality: 30
Avg. Density : 30

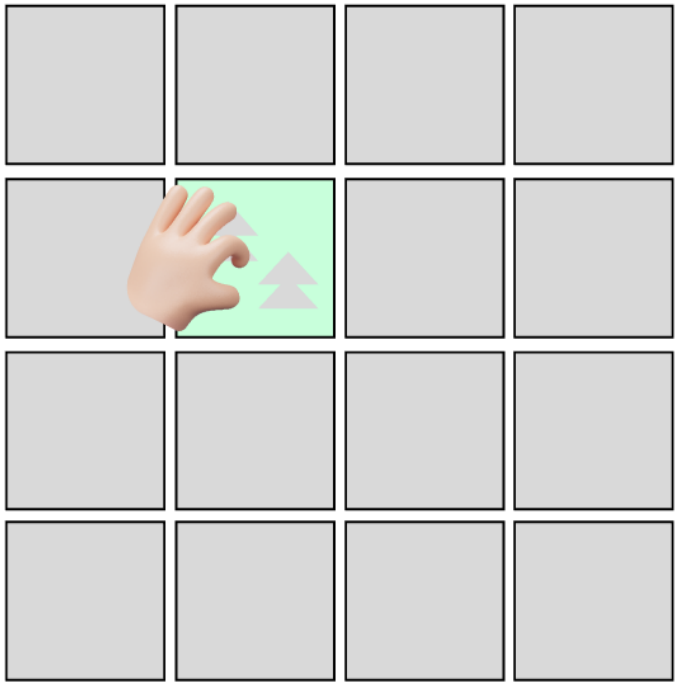
Env Score:

30

Personal: 0
Combined: 24

Personal: 0
Combined: 15

Personal: 0
Combined: 6



Environment Parameters

Avg. Greenery : 30
Avg. Vitality: 30
Avg. Density : 30

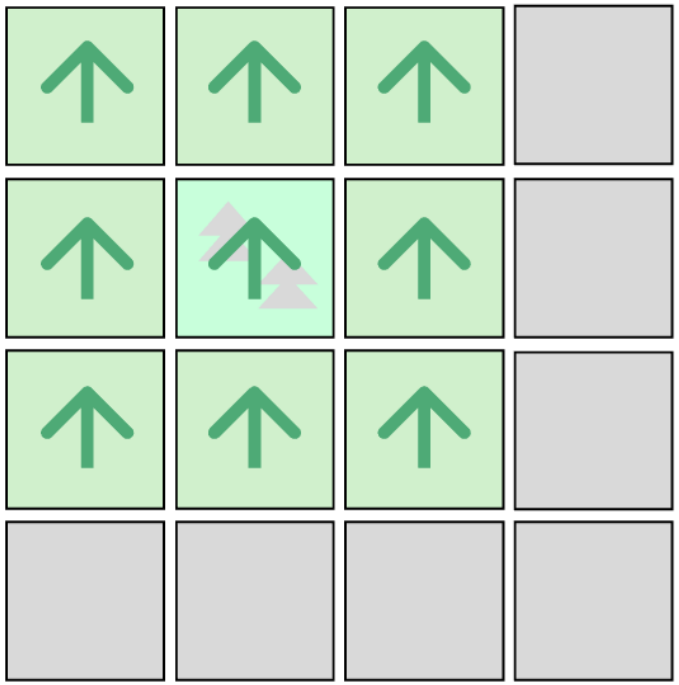
Env Score:

35

Personal: 12
Combined: 30

Personal: 10
Combined: 20

Personal: 8
Combined: 8



Environment Parameters

Avg. Greenery : 30
Avg. Vitality: 30
Avg. Density : 30

Env Score:

35

Personal: 12
Combined: 30

Personal: 10
Combined: 20

Personal: 8
Combined: 8

G: 40 V: 30 D: 20	G: 40 V: 30 D: 20	G: 40 V: 30 D: 20	
G: 40 V: 30 D: 20	G: 50 V: 30 D: 10	G: 40 V: 30 D: 20	
G: 40 V: 30 D: 20	G: 40 V: 30 D: 20	G: 40 V: 30 D: 20	

Environment Parameters

Avg. Greenery : 30
Avg. Vitality: 30
Avg. Density : 30

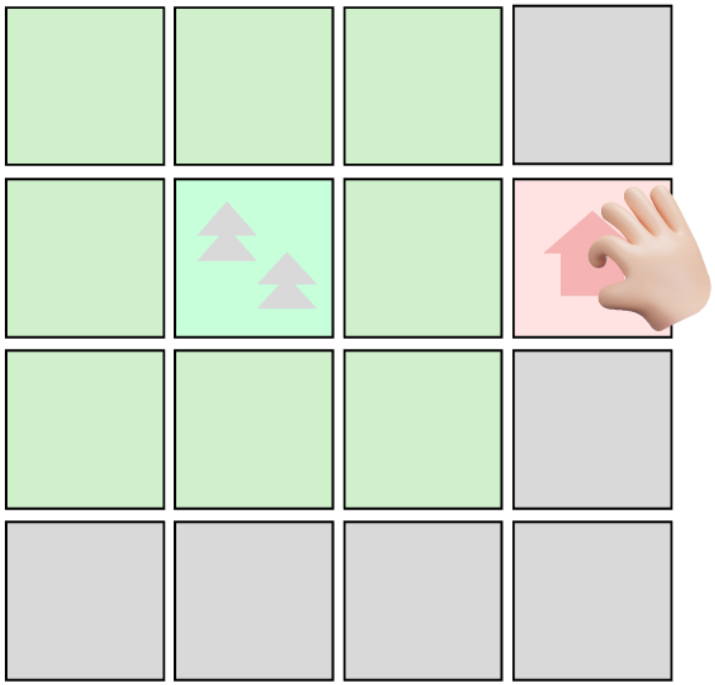
Env Score:

35


Personal: 12
Combined: 30


Personal: 10
Combined: 20


Personal: 8
Combined: 8



Environment Parameters

Avg. Greenery : 30
Avg. Vitality: 30
Avg. Density : 30

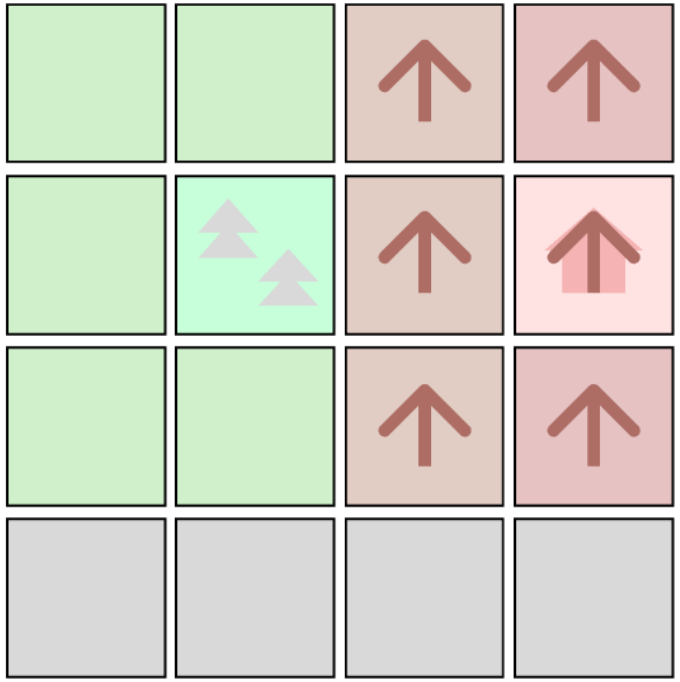
Env Score:

38

Personal: 10
Combined: 26

Personal: 12
Combined: 24

Personal: 20
Combined: 20



Environment Parameters

Avg. Greenery : 30
Avg. Vitality: 30
Avg. Density : 30

Env Score:

38


Personal: 10
Combined: 26


Personal: 12
Combined: 24


Personal: 20
Combined: 20

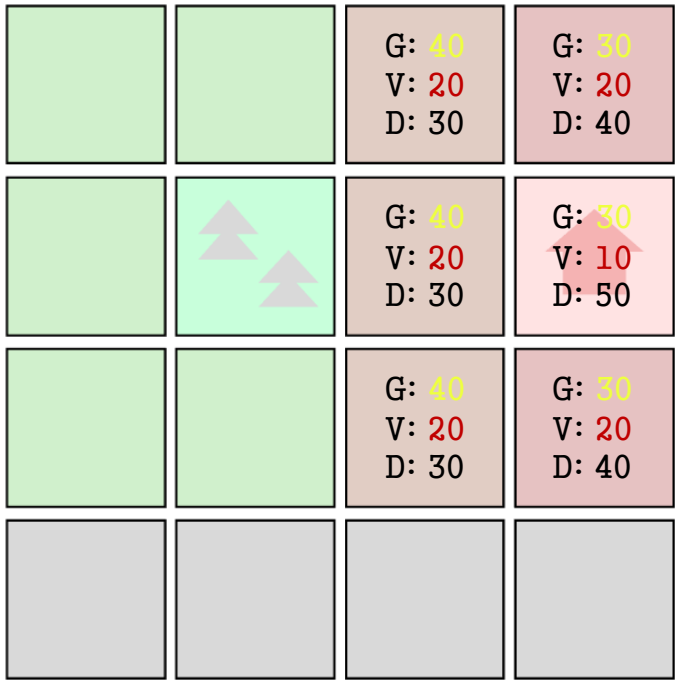


Figure 4: Turn Based Utilities

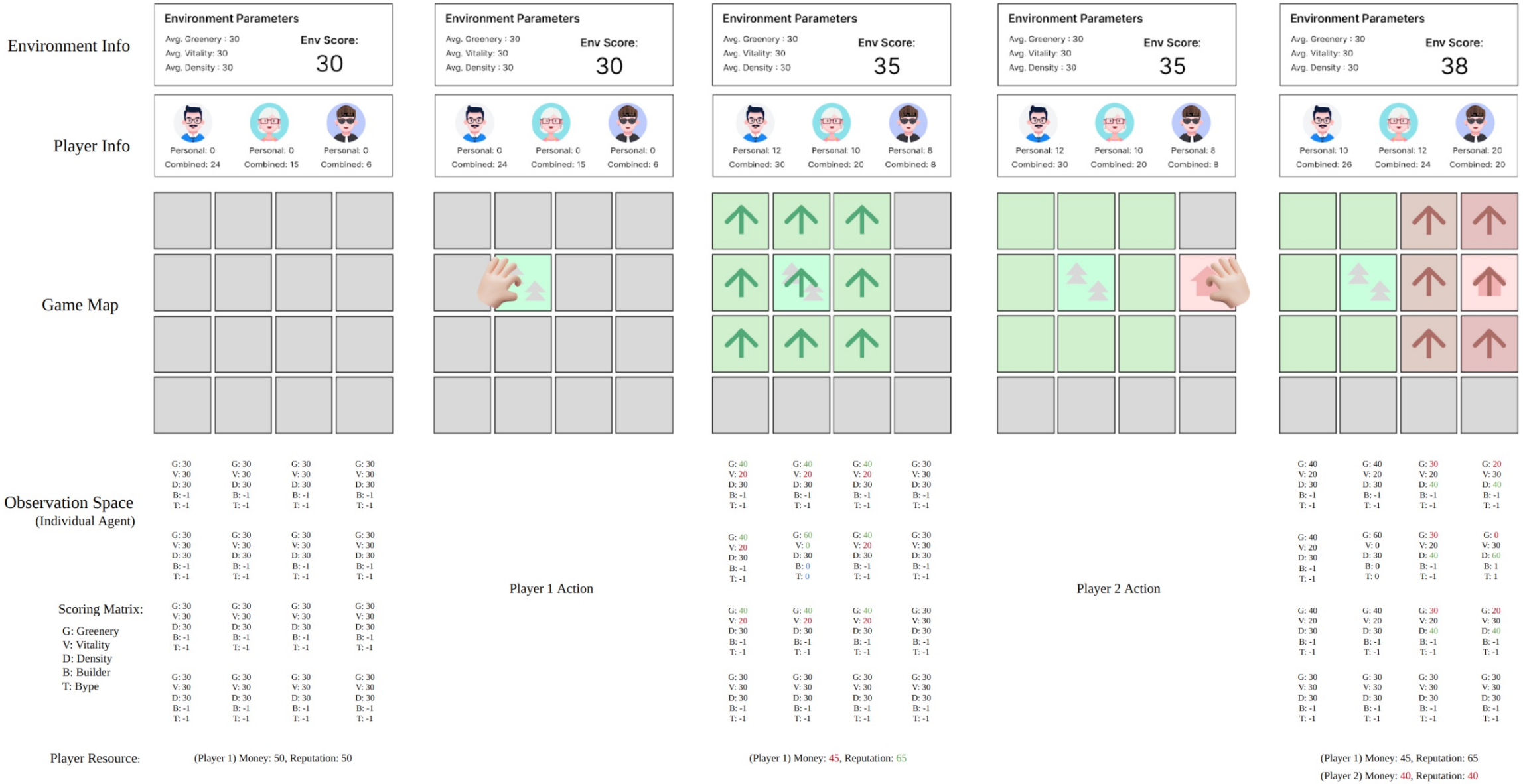


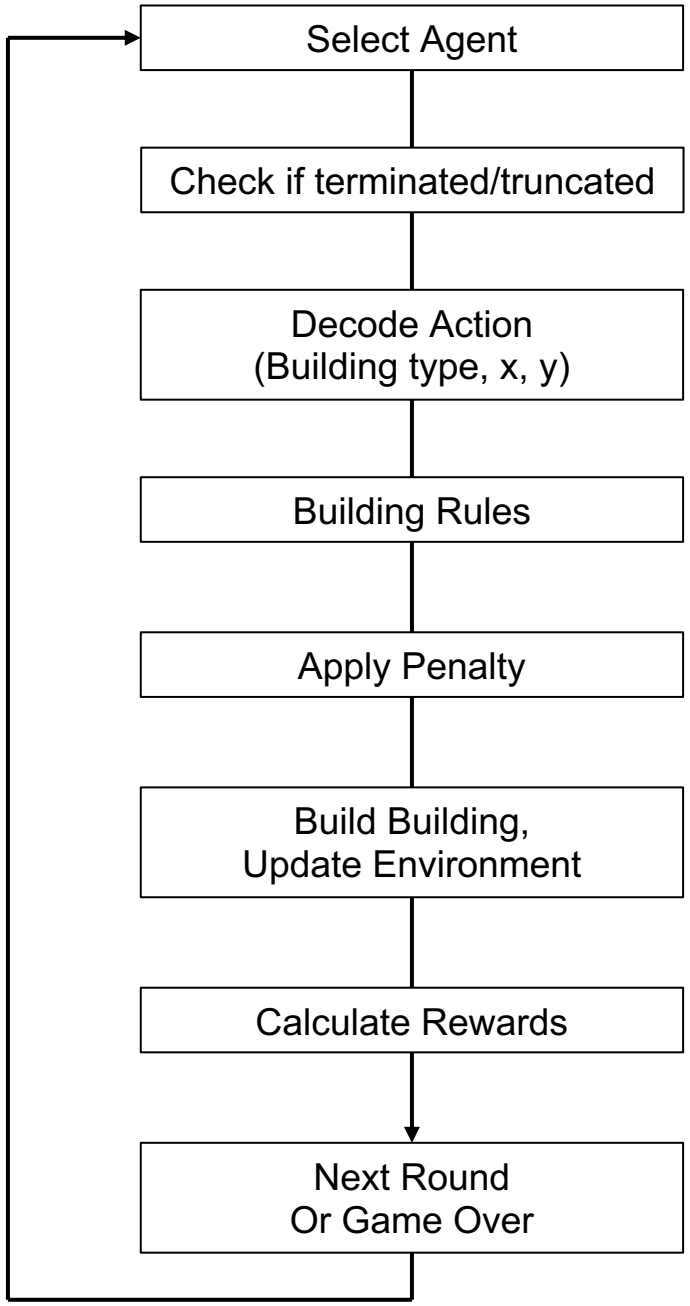
Figure 7: Observation Space for Agents

Grid Observations:
$$\left[\begin{bmatrix} -35 & 70 & 0 \\ -15 & 50 & 20 \\ 5 & 30 & 30 \\ 35 & 0 & 20 \end{bmatrix} \begin{bmatrix} -45 & 80 & -10 \\ -45 & 50 & 30 \\ 15 & 20 & 30 \\ 5 & 30 & 0 \end{bmatrix} \begin{bmatrix} -55 & 90 & 0 \\ -35 & 70 & 20 \\ -15 & 50 & 30 \\ 25 & 10 & 30 \end{bmatrix} \begin{bmatrix} -35 & 40 & 40 \\ -35 & 70 & 0 \\ 5 & 30 & 30 \\ -5 & 10 & 60 \end{bmatrix}\right]$$

Builder Locations:
$$\begin{bmatrix} 0 & -1 & -1 & 2 \\ 0 & 2 & -1 & 1 \\ 0 & -1 & -1 & 2 \\ 1 & 1 & -1 & 1 \end{bmatrix}$$

Building Types:
$$\begin{bmatrix} 2 & -1 & -1 & 0 \\ 2 & 1 & -1 & 2 \\ 2 & -1 & -1 & 0 \\ 1 & 2 & -1 & 1 \end{bmatrix}$$

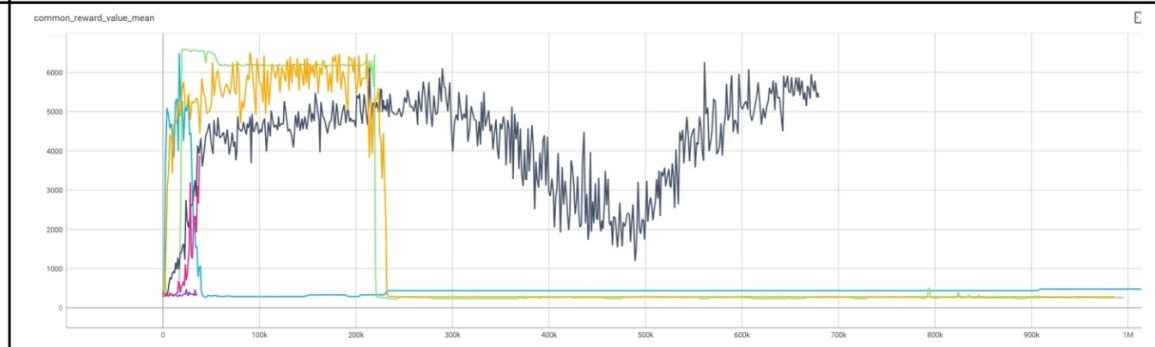
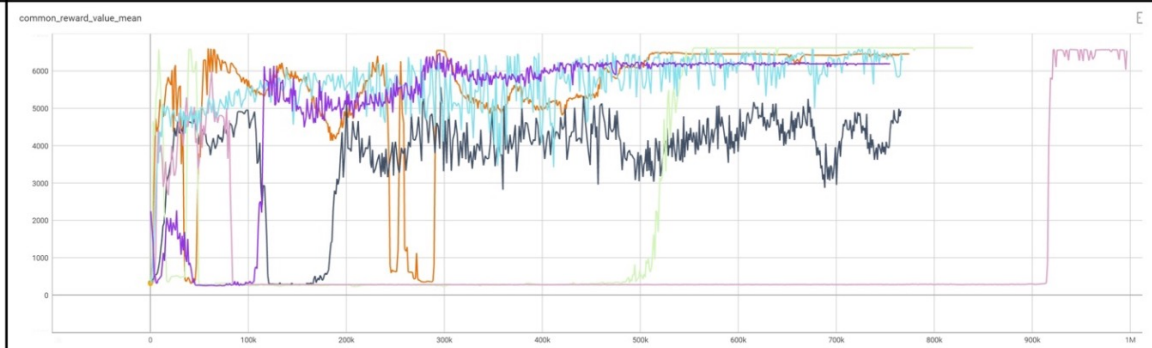
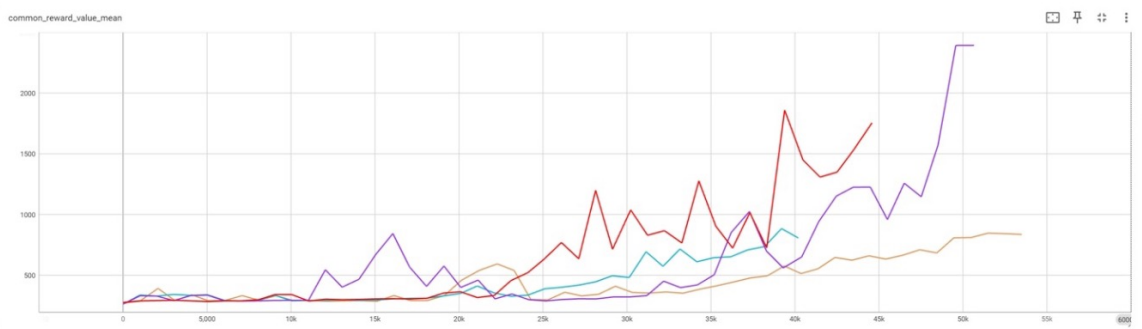
Player Resources:
$$\begin{bmatrix} \text{Money: } 84 \\ \text{Reputation: } 54 \end{bmatrix}$$



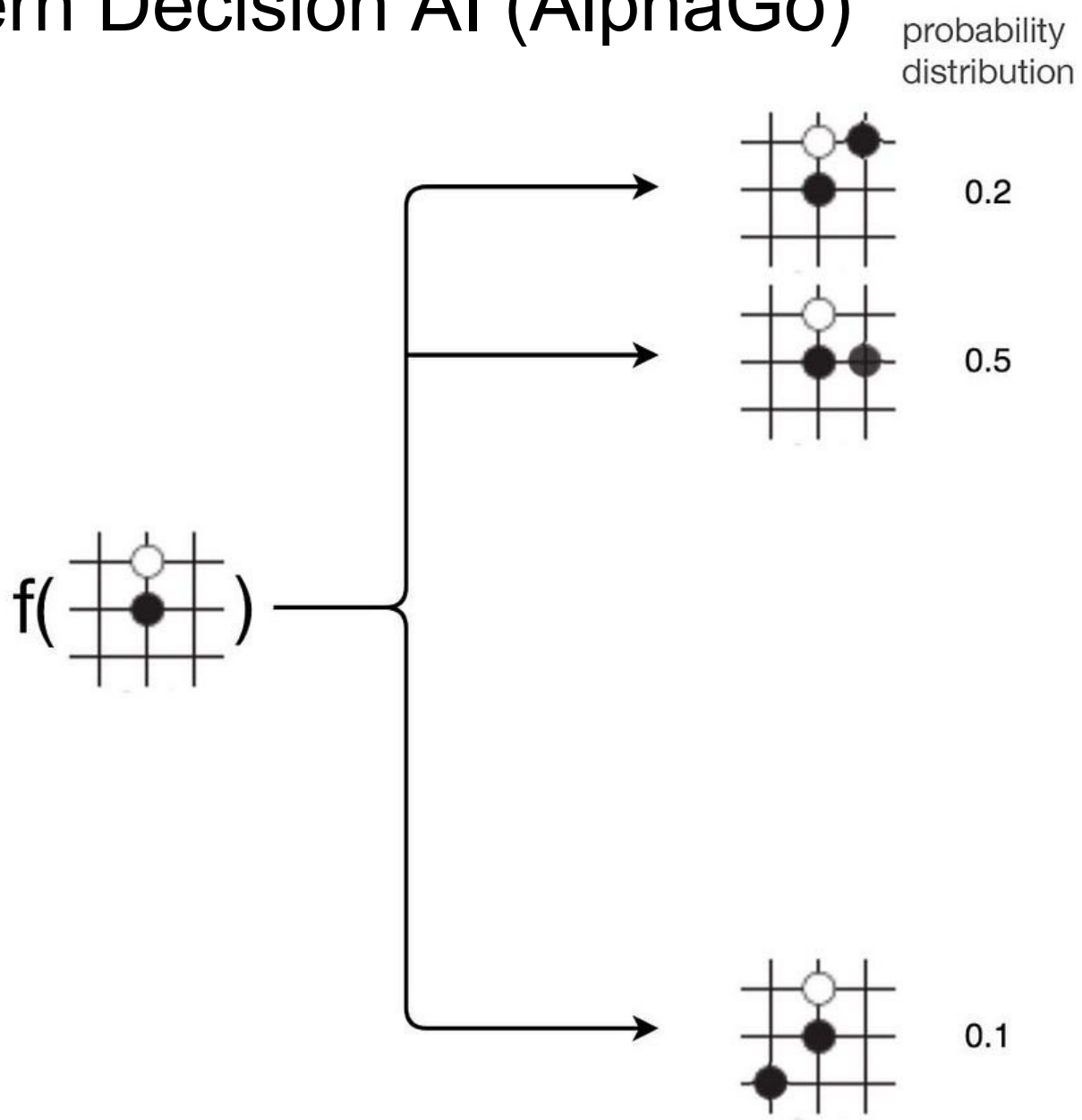
Algorithm 1: Step Function in SimCity Environment

Input
  Current Agent Action *action*, Training Mode (maximize individual reward or common reward)
Output
  Updated Environment State(obs), Rewards, Terminated, Truncated, and Info

Initialize *agent* ← self.agent_selection;
if *self.terminations[agent] or self.truncations[agent]* **then**
  return;
Decode Action: $(building\_type, x, y) \leftarrow$ decode_action(action);
if $action = NO\_OP$ **then**
  No operation performed by agent;
else
  if *self.buildings[x][y] is not None* **then**
    Apply penalty for building on occupied cell;
    $reward \leftarrow reward +$ build_on_occupied_penalty;
  else
    if *Agent lacks resources to build* **then**
      Apply penalty for insufficient resources;
      $reward \leftarrow reward +$ not_enough_resource_penalty;
    else
      Deduct resources from agent;
      Update environment with new building;
      Adjust grid and neighboring cells based on building effects;
      $reward \leftarrow reward +$ immediate_reward;
Update Utilities: Adjust player scores and resources based on building utilities;
Calculate Environment Score: $env\_score \leftarrow$ calculate_environment_score();
Determine Player Reward Weights Based on Type : $(\alpha, \beta) \leftarrow$ weights based on player type;
Update Integrated Score:
self.players[agent].integrated_score $\leftarrow \alpha \times$ self.players[agent].self_score $+ \beta \times env\_score$;
Log Rewards: Compute individual reward for *agent* and common reward as the sum of individual reward.
Update Rewards: **if** *mode = maximize individual reward* **then**
  $reward \leftarrow$ each agent's reward (tuple);
else
  $reward \leftarrow$ common reward (float);
Increment Move Counter: self.num_moves ← self.num_moves + 1;
Check Termination Condition: `self.terminations` updated if game is over;
Select Next Agent: self.agent_selection ← self._agent_selector.next();

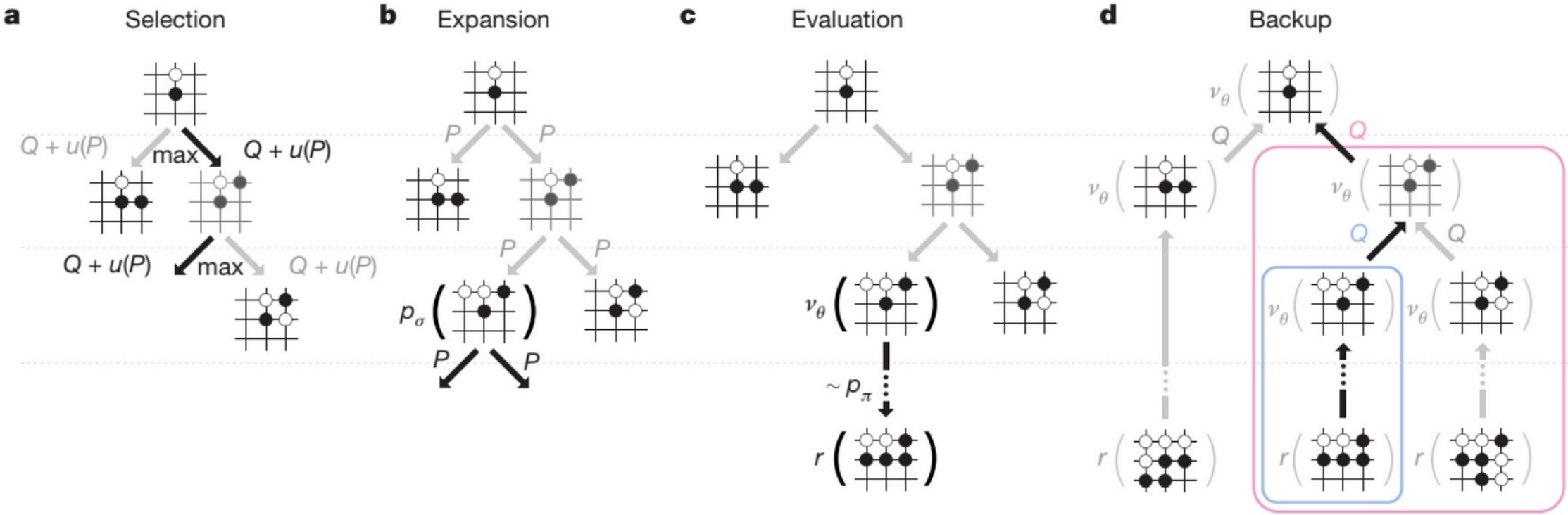
Category	Graph	Legend
Independent Learning		Actor-Critic (IA2C) – No Shared Information Actor-Critic (IA2C) –Shared Information IPPO – No Shared Information IPPO – Shared Information Q-Learning (IQL) – No Shared Information Q-Learning (IQL) – Shared Information
Centralized Critic		COMA – No Shared Information COMA – Shared Information MAA2C – No Shared Information MAA2C – Shared Information MAPPO – No Shared Information MAPPO – Shared Information
Value Decomposition		QMIX – No Shared Information QMIX – Shared Information VDN – No Shared Information VDN – Shared Information

Reflect on Modern Decision AI (AlphaGo)



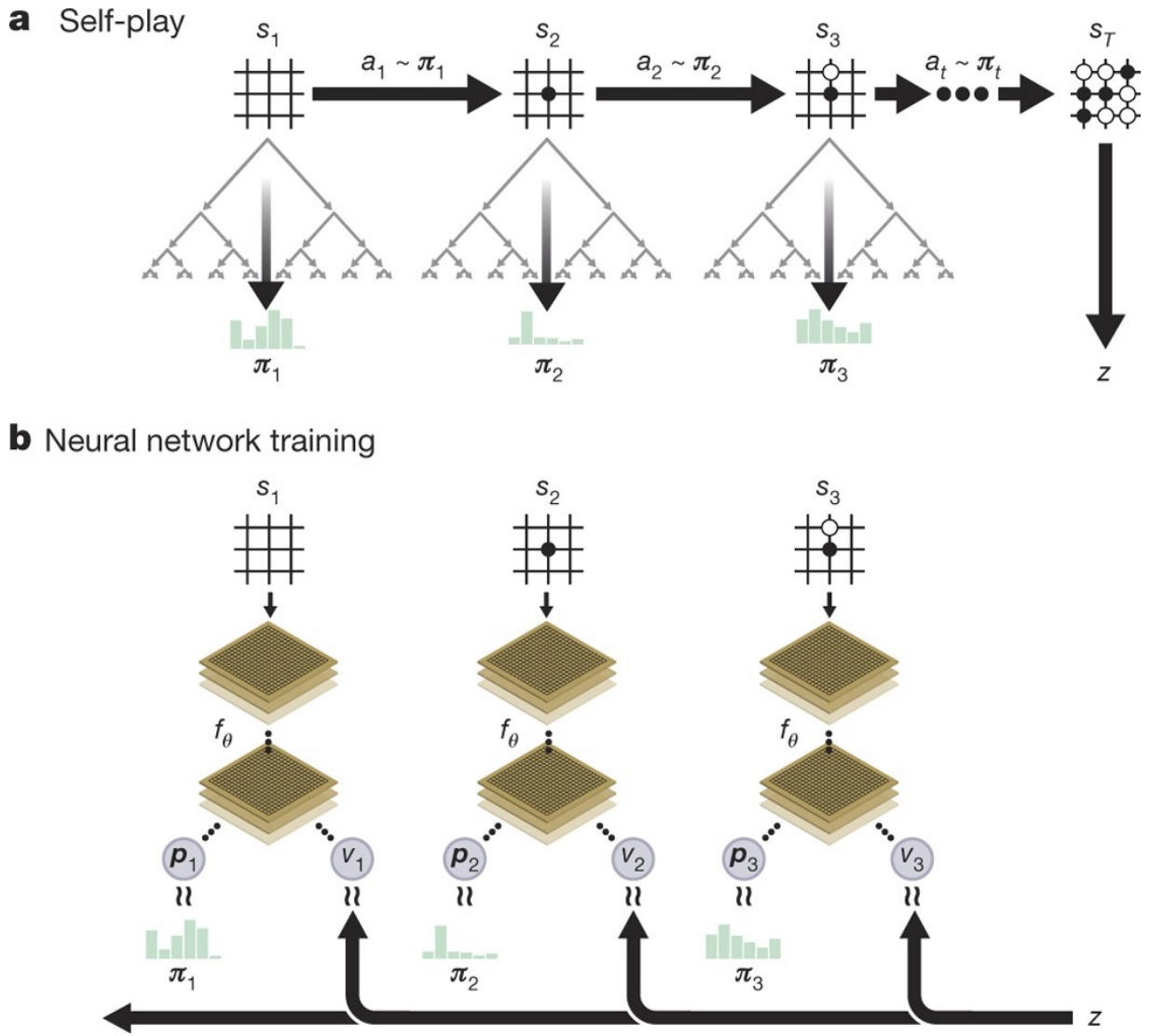
https://jonathan-hui.medium.com/alphago-how-it-works-technically-26ddcc085319

Reflect on Modern Decision AI (AlphaGo)



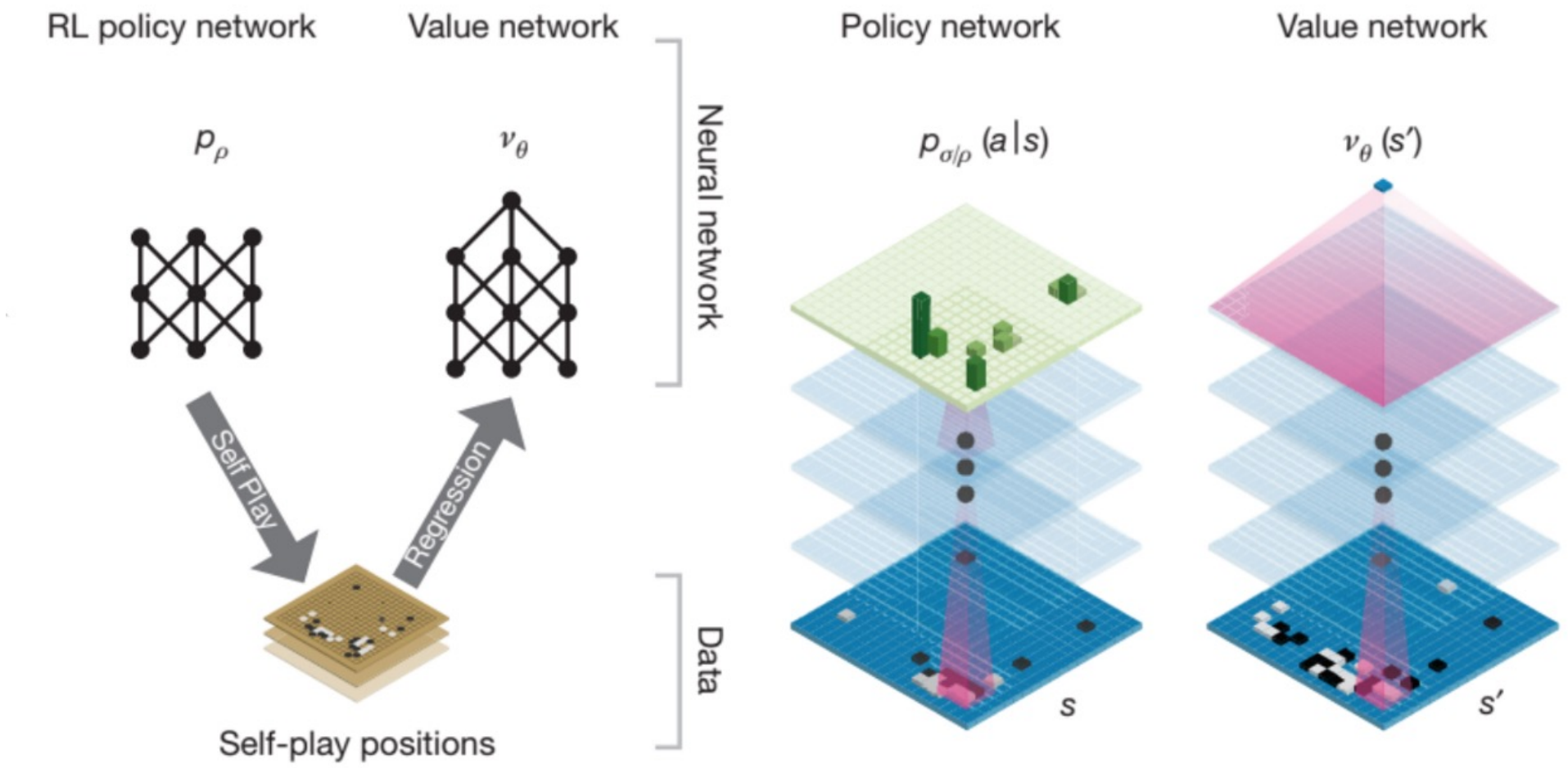
https://jonathan-hui.medium.com/alphago-how-it-works-technically-26ddcc085319

Reflect on Modern Decision AI (AlphaGo)



https://jonathan-hui.medium.com/alphago-how-it-works-technically-26ddcc085319

Reflect on Modern Decision AI (AlphaGo)



https://jonathan-hui.medium.com/alphago-how-it-works-technically-26ddcc085319

Design as a Game

Thank you for joining us!

Designing Consensus:

Gamified Modeling and Simulation of Collaborative Decision-Making

Instructor: Jin Gao

IAP 2025 (Non-Credit) gaojin@mit.edu

SCAN ME

Thank you for joining us!