FAKULTÄT FÜR MATHEMATIK, INFORMATIK UND STATISTIK
INSTITUT FÜR INFORMATIK

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LEHRSTUHL FÜR DATENBANKSYSTEME UND DATA MINING

Contact-Averse Reinforcement Learning

KDDone

David Dodel, Jan Alexander Jensen,

Matthias Kraus, Xingjian Chen

2020-07-22

DBS Supervisor: Sabrina Friedl

Agenda

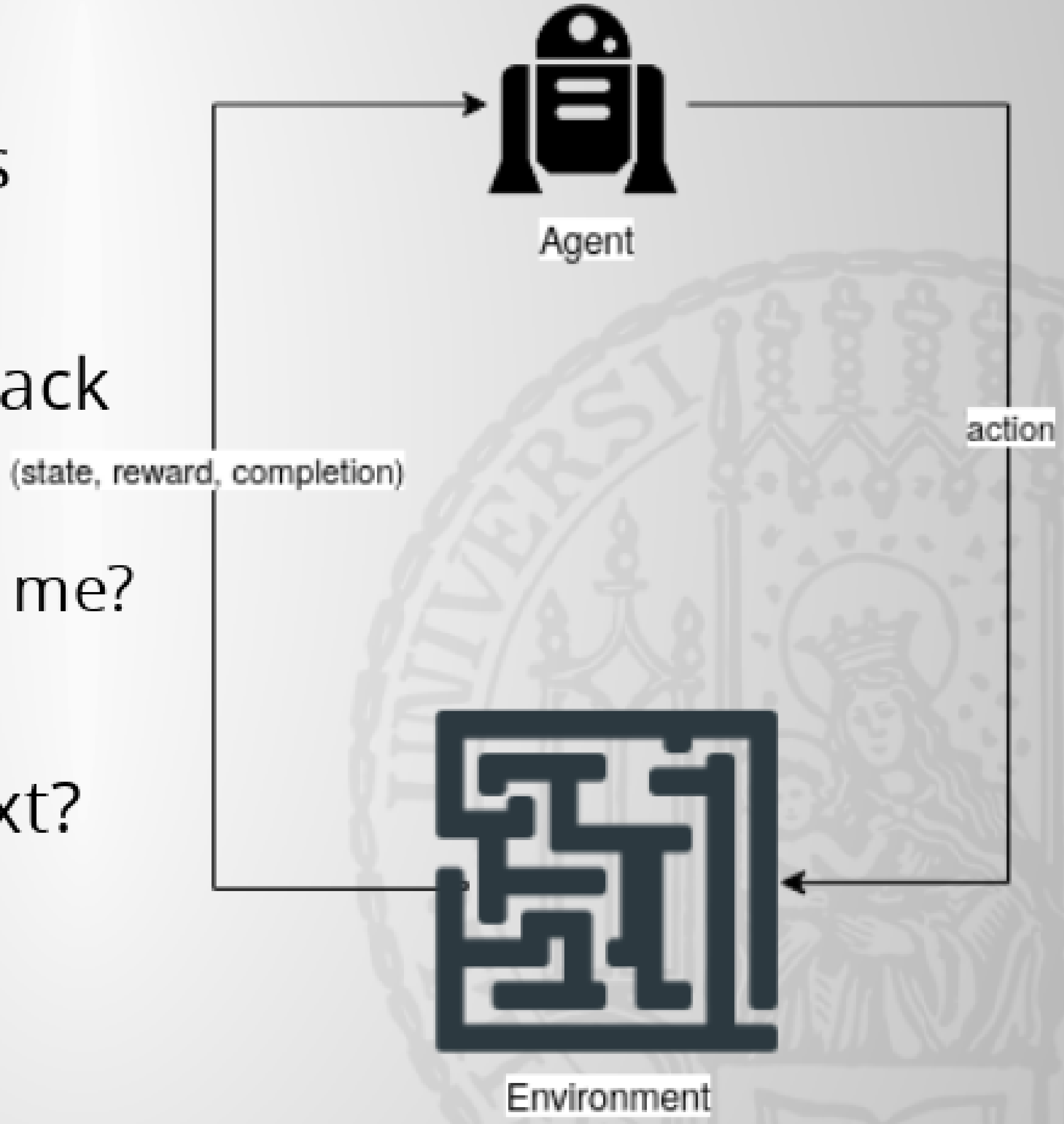
1. Project objective
2. Theoretical Background
3. Software Stack
4. Development Process
5. Simulation Environment
6. Agents
7. Evaluation
8. Lessons Learned

Problem Statement

- Context: COVID-19 outbreak
- Contact-Averse Reinforcement Learning
  - Social Distancing
  - Reinforcement Learning
- Can we train self-learning agents to avoid contact while still achieving their goals?
- Bonus: Can we reproduce behaviors seen in real-life human agents within simulated multi-agent "games"?

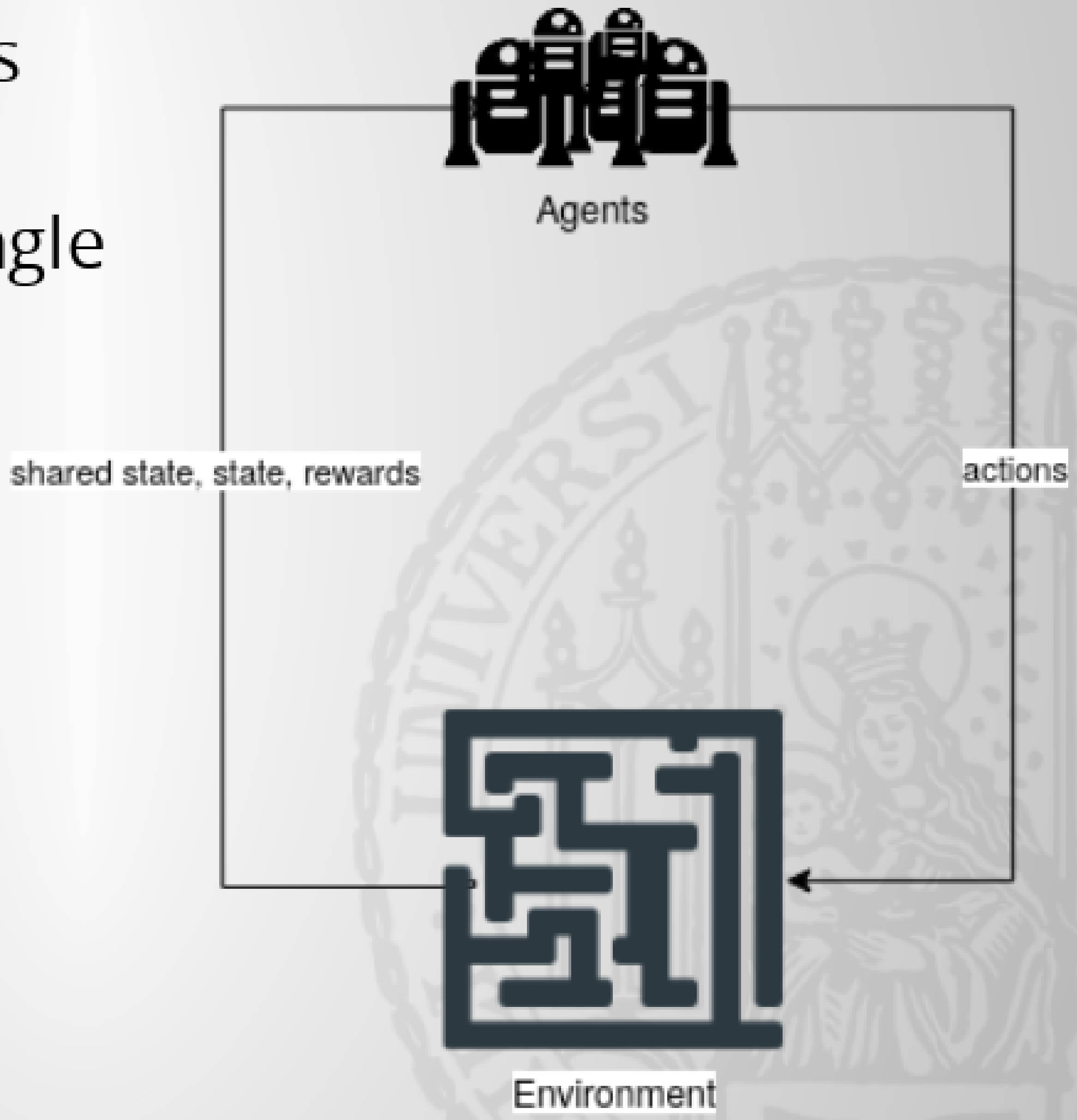
Reinforcement Learning

- (Machine) Learning
  - Self-learning algorithms
- Reinforcement
  - Learning through feedback from actions
- Observation: What's around me?
  - (agent) state
- Policy: What am I doing next?
- Reward: Was it good/bad?

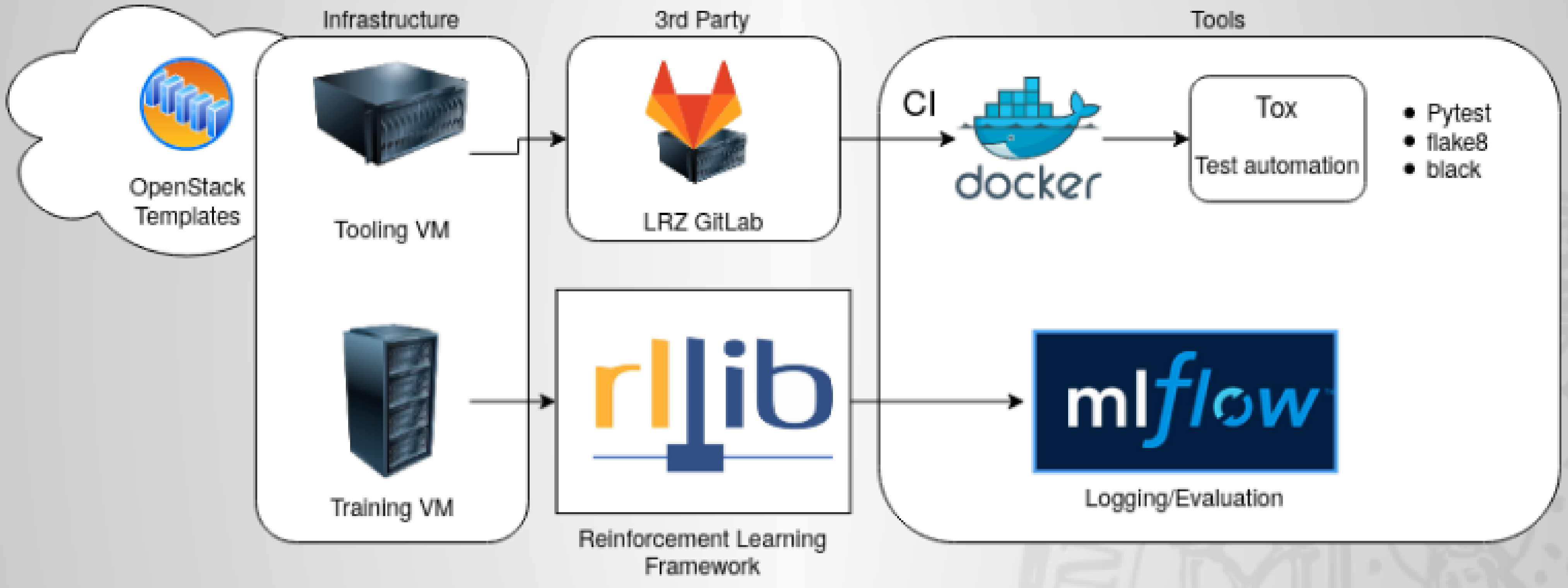


Multi-agent RL (MARL)

- Interesting to model games and interactions
  Not trivial to scale from single
- agent
  - n x agents - 1 x env
- Approaches
  - Share state
  - Share policy
  - Communicate
  - Compose policies
  - ...



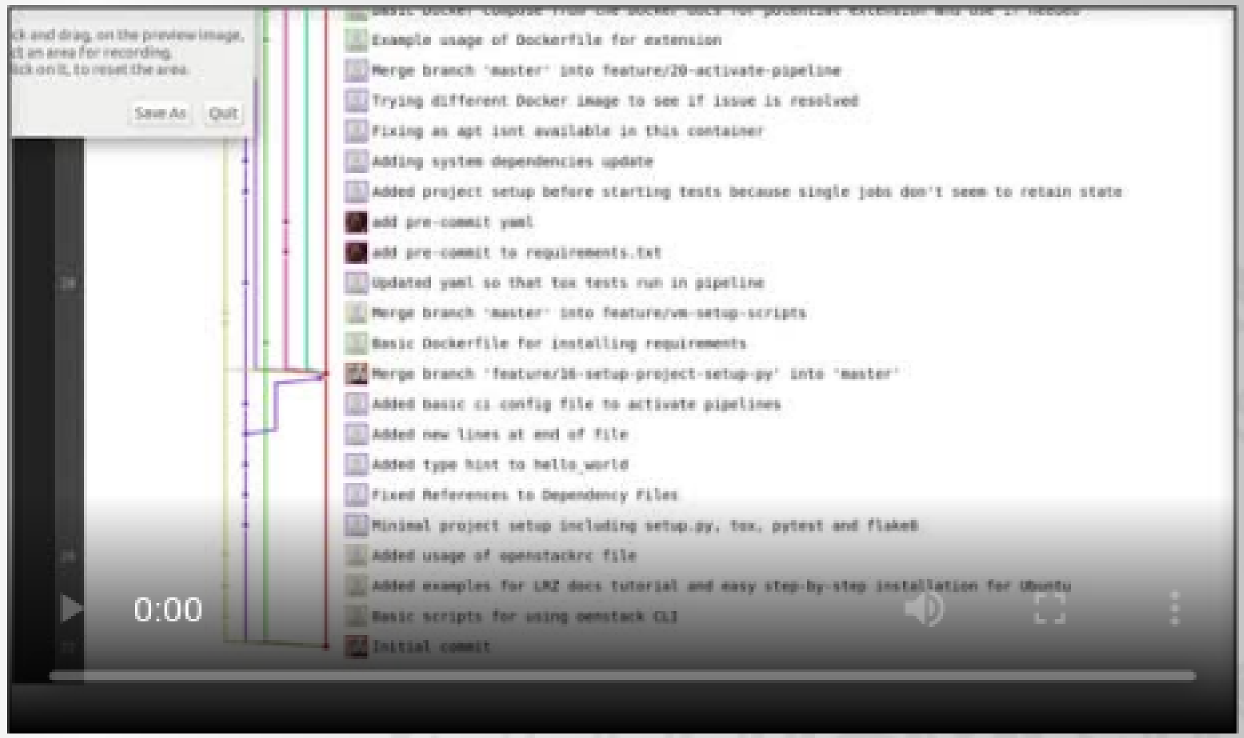
Software Stack



- OpenStack: VMs and infrastructure
- Tox: Test automation
- RLlib: Training/Tuning and algorithms
- MLflow: Logs and Evaluation

Software Development

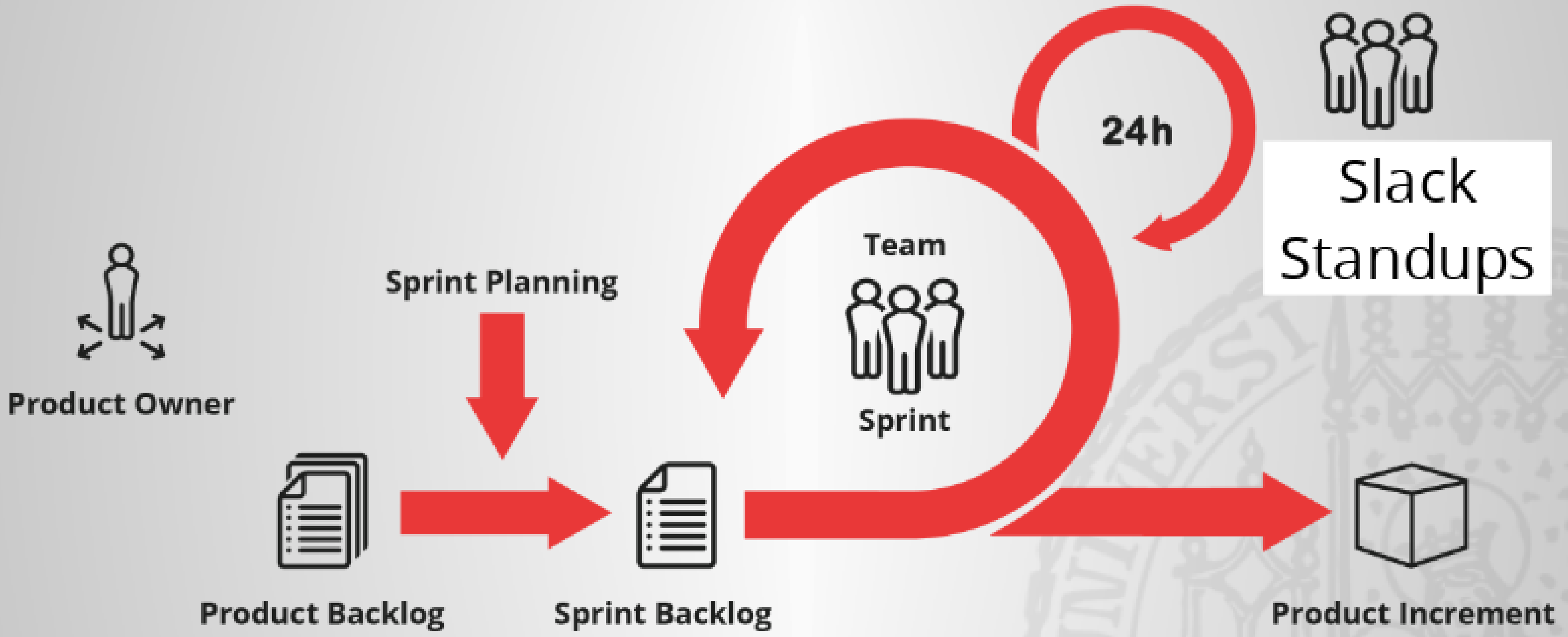
```
tests
├── __init__.py
├── logging_output_DQN_dijkstra.py
├── test_agent_state.py
├── test_CustomMetrics.py
├── test_CustomPolicy.py
├── test_Dijkstra.py
├── test_MultiGrid.py
├── test_NESW.py
├── test_render_viz.py
└── test_reward_function.py
```



- Test Driven Development
- GitLab Flow => slight adjustments

- LoC: 4240
- Issues: 74

Software Development



- Protocols and Wiki for tracking
- Weekly Zoom standups
- Iterate on process

Sprint recap

Sprint 0

- Infrastructure Setup
- Tooling

Sprint 1

- Infrastructure Setup
- Paper Research
- Implementation Concept

Sprint 2

- Environment
- ML Flow Logging
- Random Sampling Agent

Sprint 3

- Environment Visualization
- Shortest Path Agent

Sprint 4

- Configurable Observations
- DQN Agent

Sprint 5

- Scenario Implementation
- Evaluation

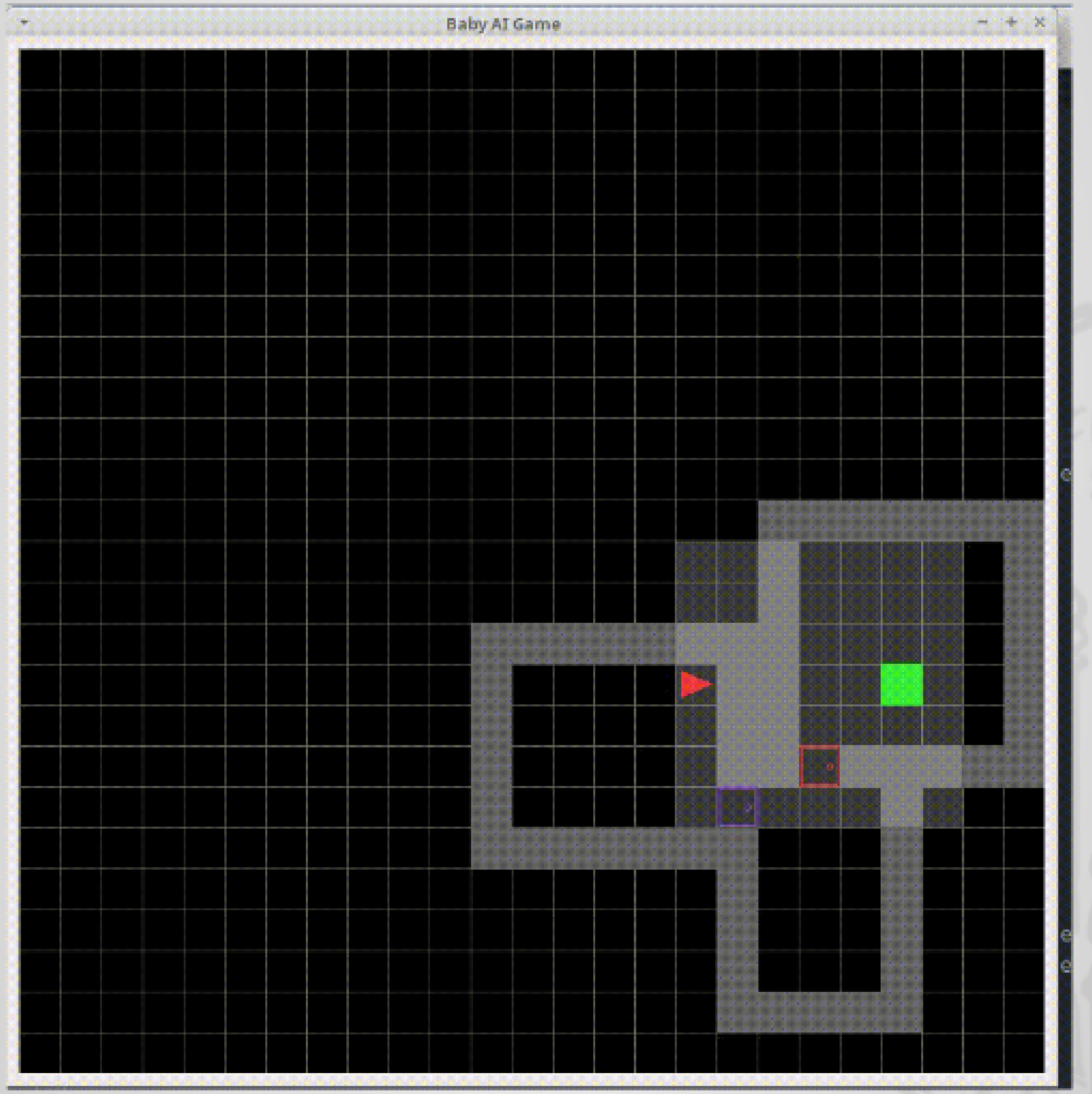
Sprint 6

- Finalization
- KDDone :)

Environment

- Based on gym-minidgrid
- Supports interaction of multiple agents
- Action Space
  - North/East/South/West
  - Wait
- Rewards
  - step/wait: -0.01
  - stepping into the same cell: -0.5
  - reaching the goal: 1.0
- Configurable Observation Space

Random Multi Room



Shortest Path Agent

- Baseline Agent
  - Get to goal without taking contact aversion into consideration
- Calculates shortest path based on Dijkstra
  - Randomized when multiple shortest paths with the same length

DQN

- ## Q- Learning
  - basic algorithm based on maximise Q value
  - Unstable when using nonlinear function approximator

$$Q(s,a) = r + \gamma \max_{a'} Q(s',a')$$

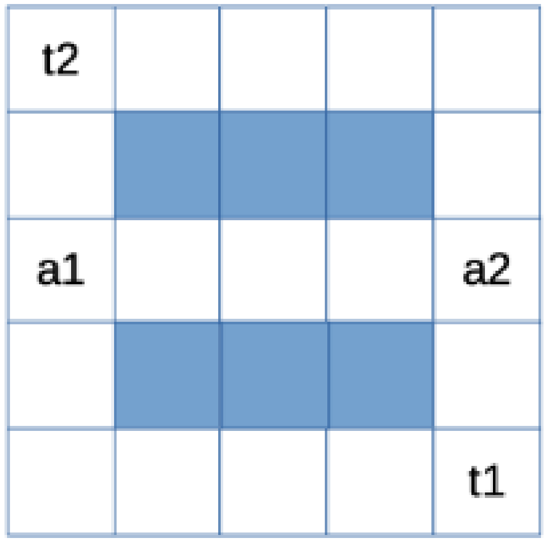
- ## Deep Q Network
  - Using Deep Neural Network to map state to action
  - Features:
    - Experience Replay <state, action, reward, next_state>
    - Target Network

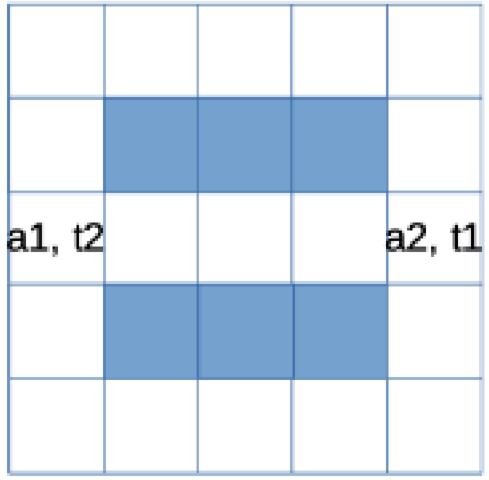
IQL

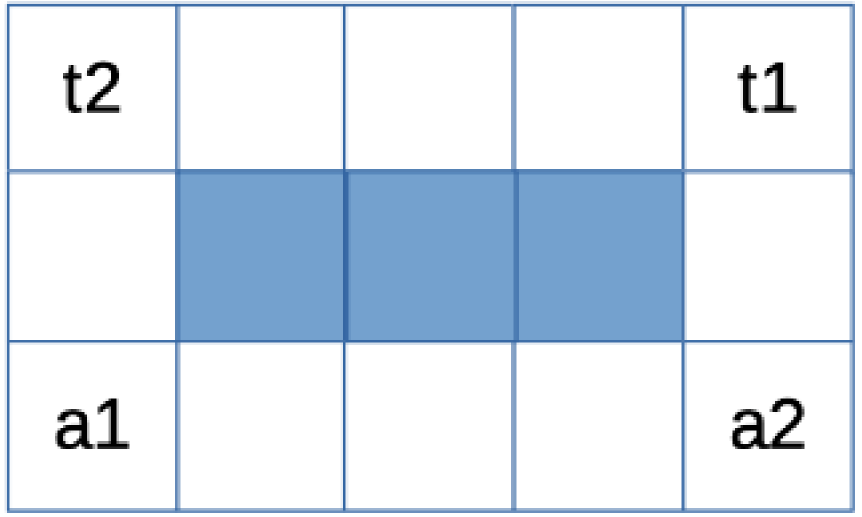
- Independent Q Learning
- Solution for solving Multi-agent Problem
- Ray support multi-agent
  - Example: shortestPathAgent+DQN

```python
def policy_mapping_fn(agent_id):
    if agent_id % 2 == 0:
        return "dijkstra_policy"
    else:
        return "dqn_policy"

...
policies = {
    "dijkstra_policy": (DijkstraPolicy, obs_space, act_space, {}),
    "dqn_policy": (DQNTorchPolicy, obs_space, act_space, {}),
}

dijkstra_trainer = DijkstraTrainer(
    env="twoWallsEnv",
    config={...},
)

dqn_trainer = DQNTrainer(
    env="twoWallsEnv",
    config={...},
)
```

Evaluation Environments

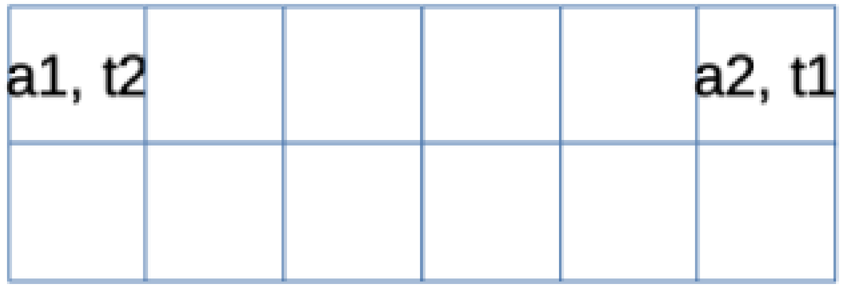






a1: agent 1, **a2**: agent 2

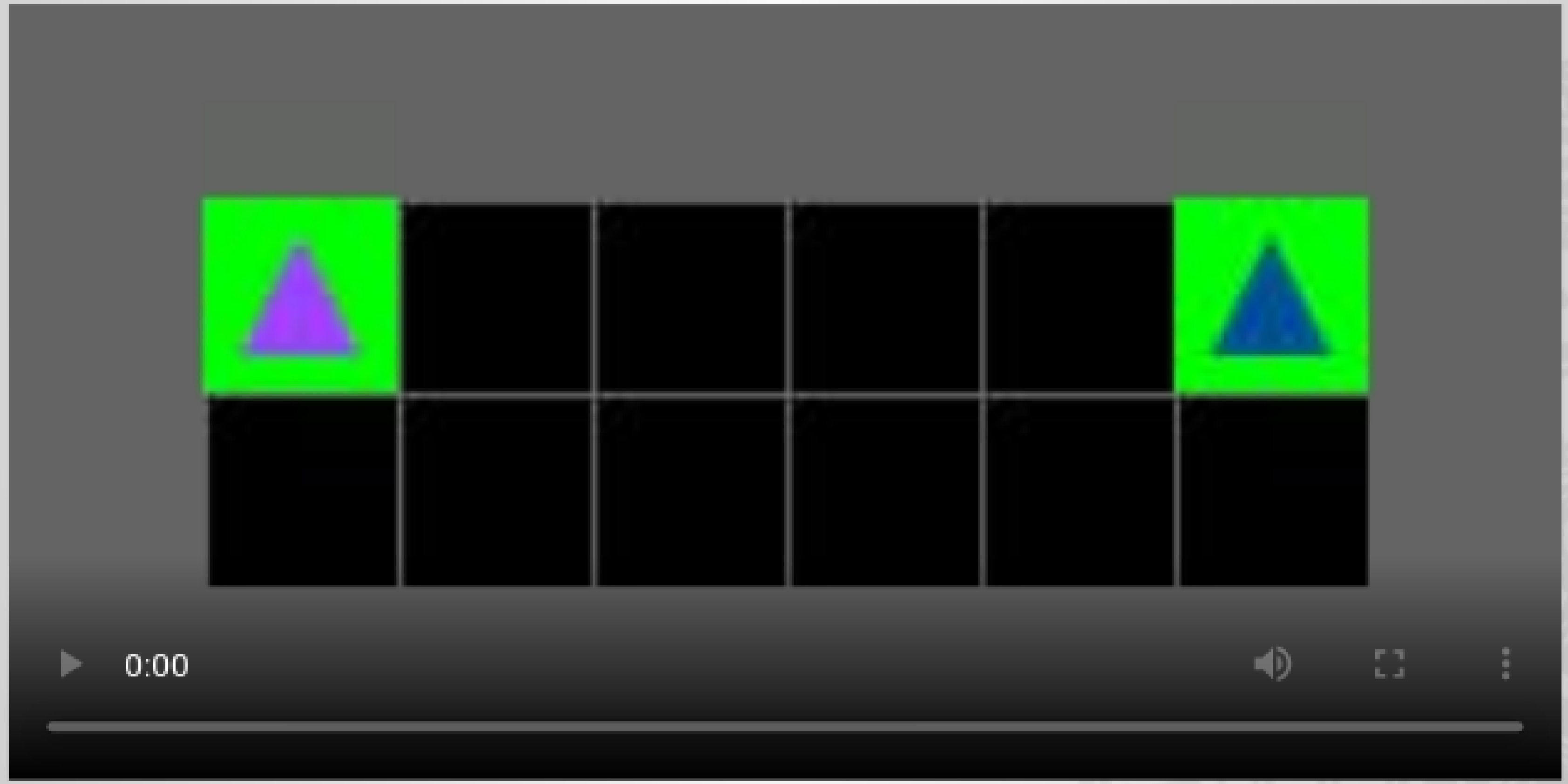
t1: target 1, **t2**: target2



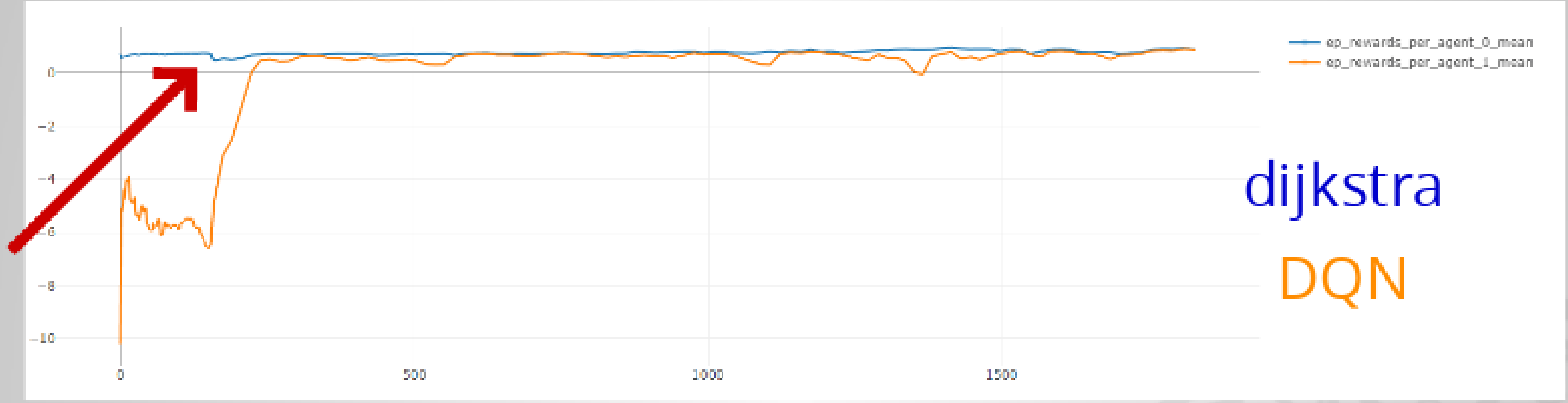
env #1

result #1 - dijkstra

reward: -0.05 - 0.5 +1 = 0.45



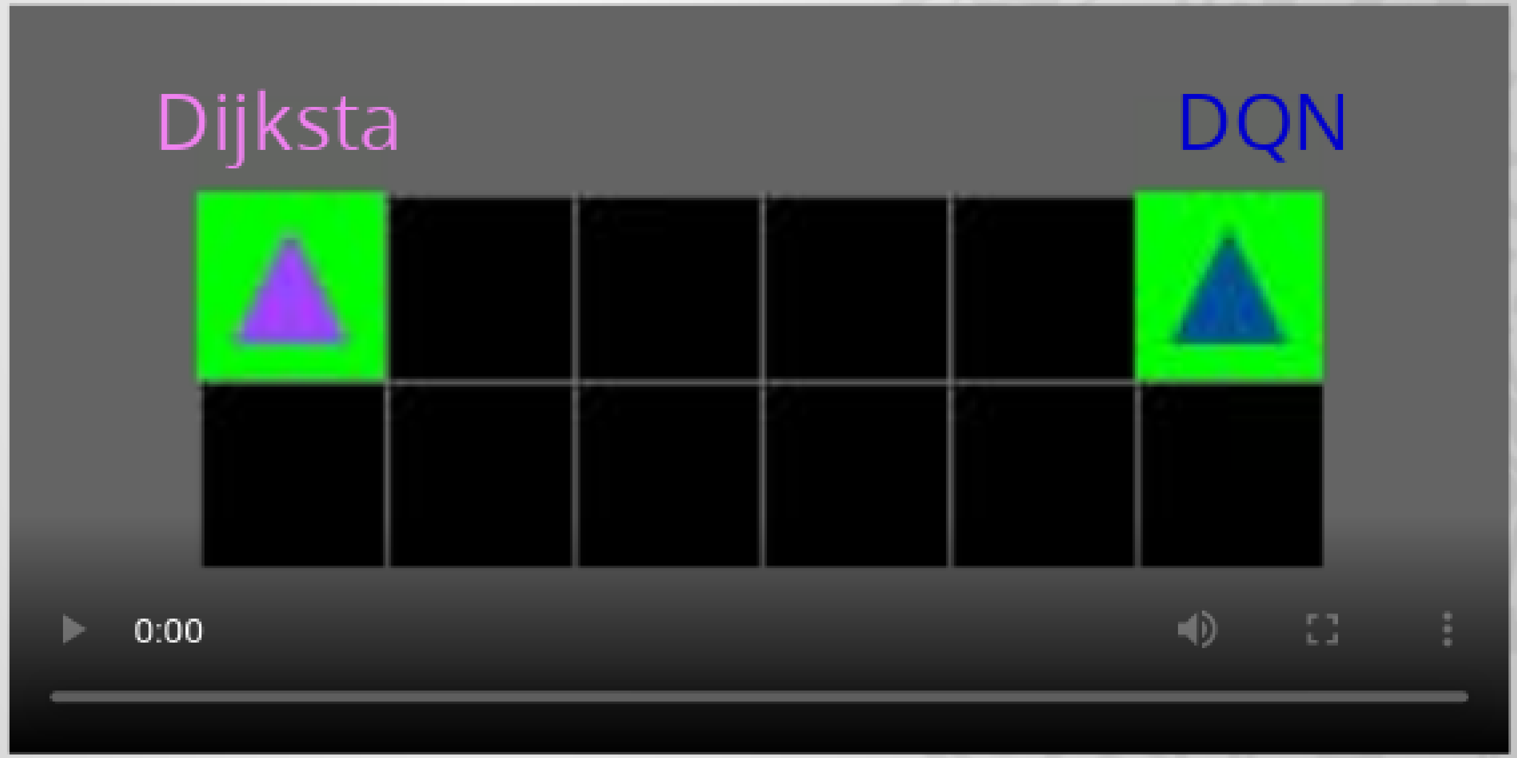
result #1 – dijkstra + DQN



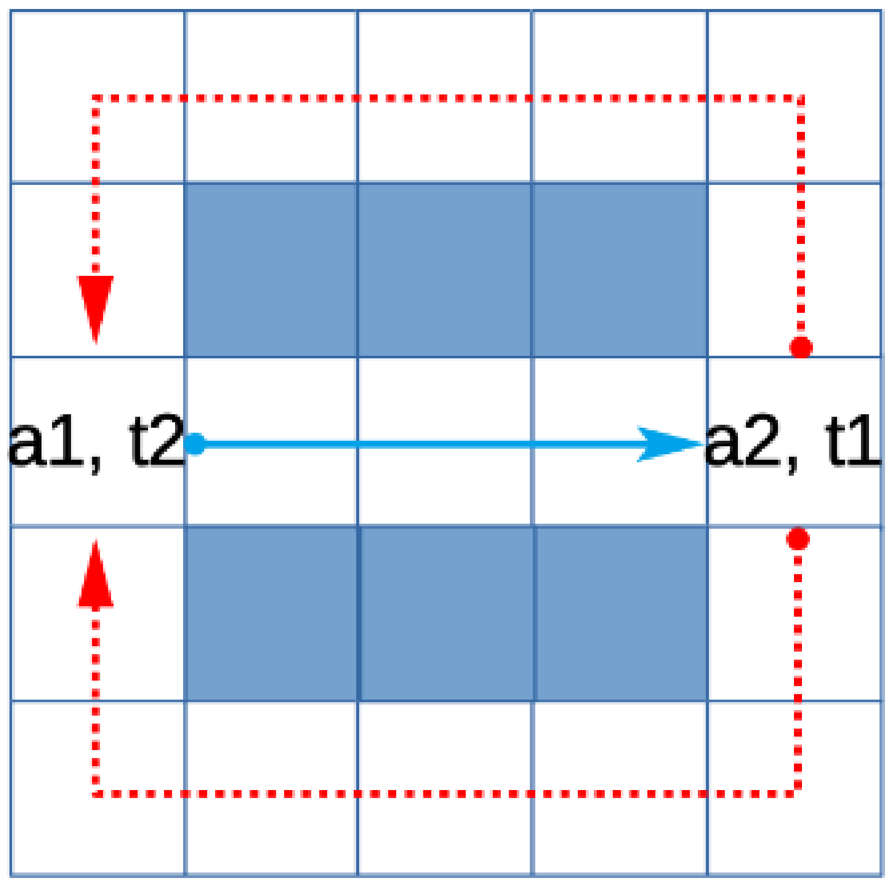
reward:

-0.05 + 1 = 0.96

-0.07 + 1 = 0.93

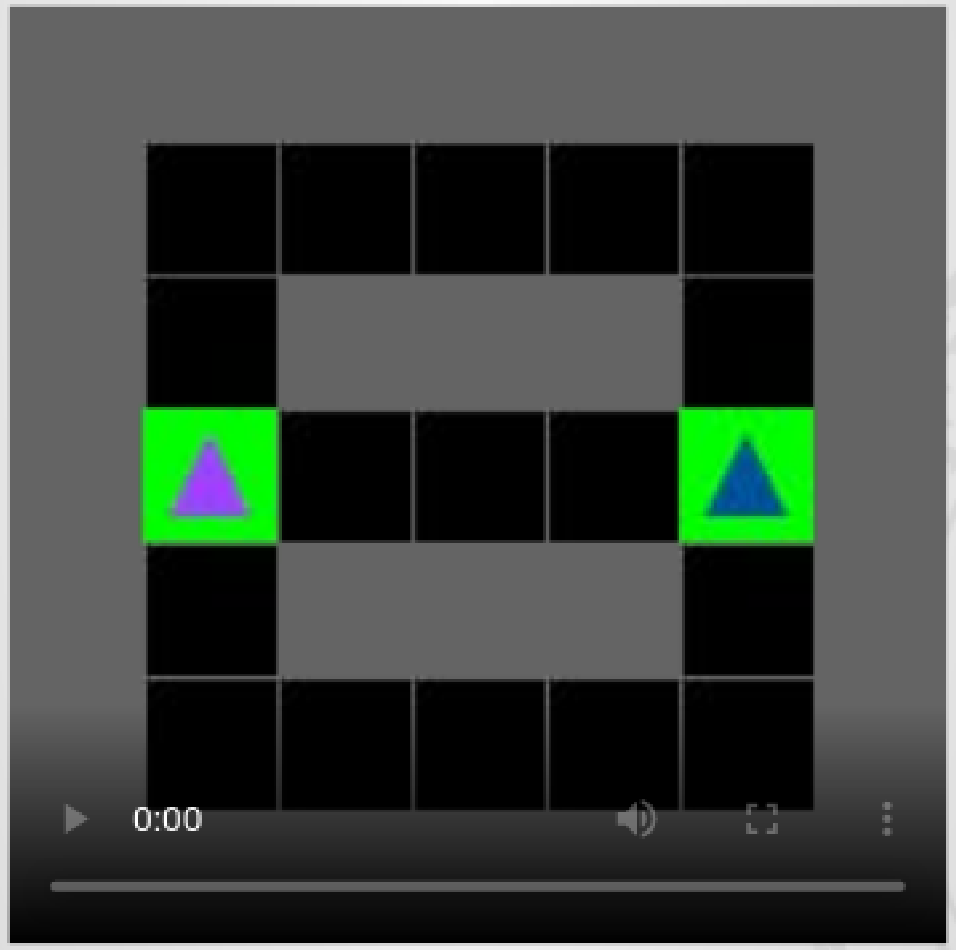


env #2



result #2 - dijkstra

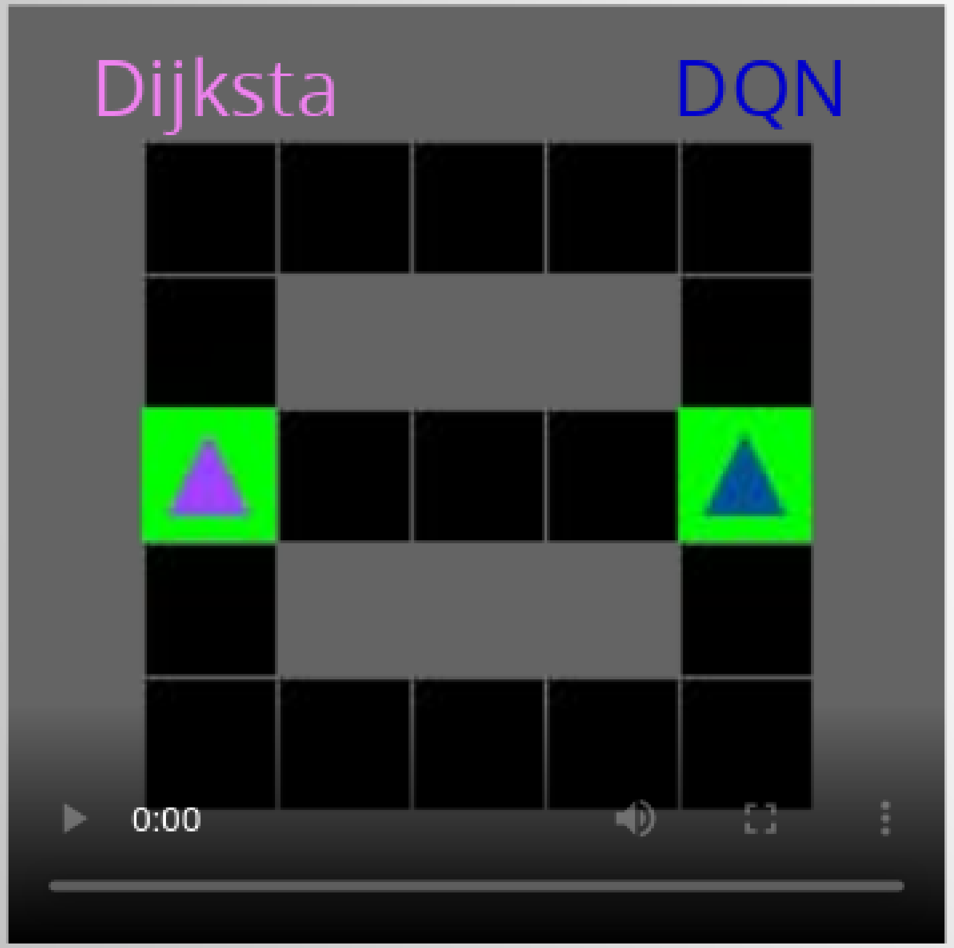
-0.04 - 0.5 + 1 = 0.56

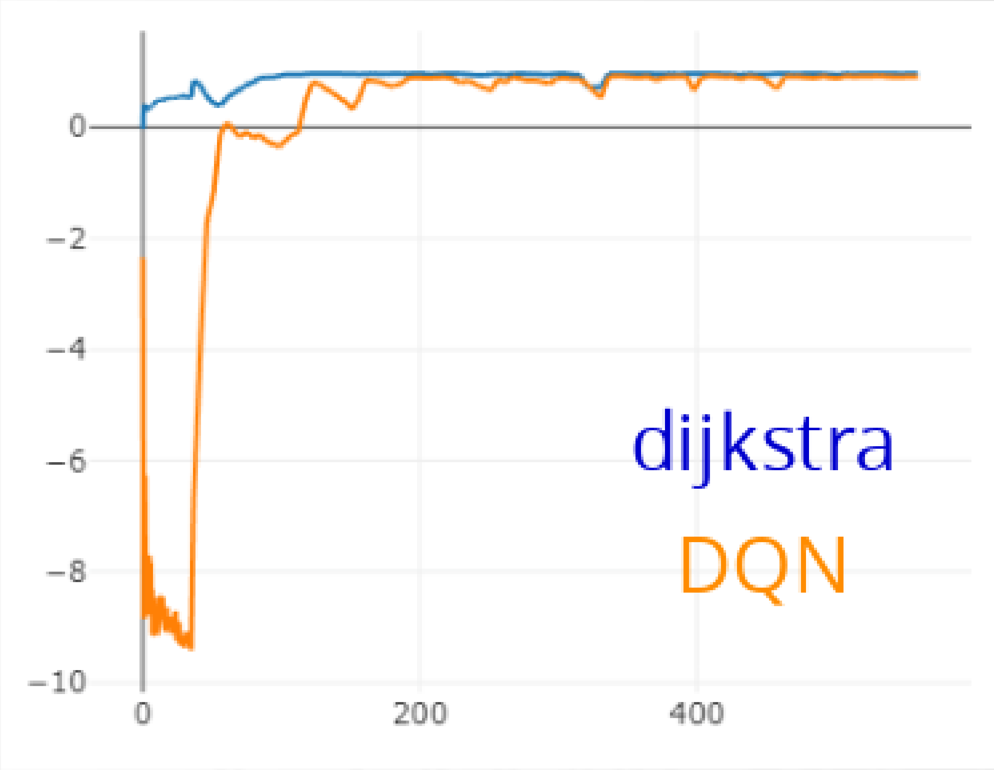


result #2 – dijkstra + DQN

-0.04 + 1 = 0.96

-0.08 + 1 = 0.92

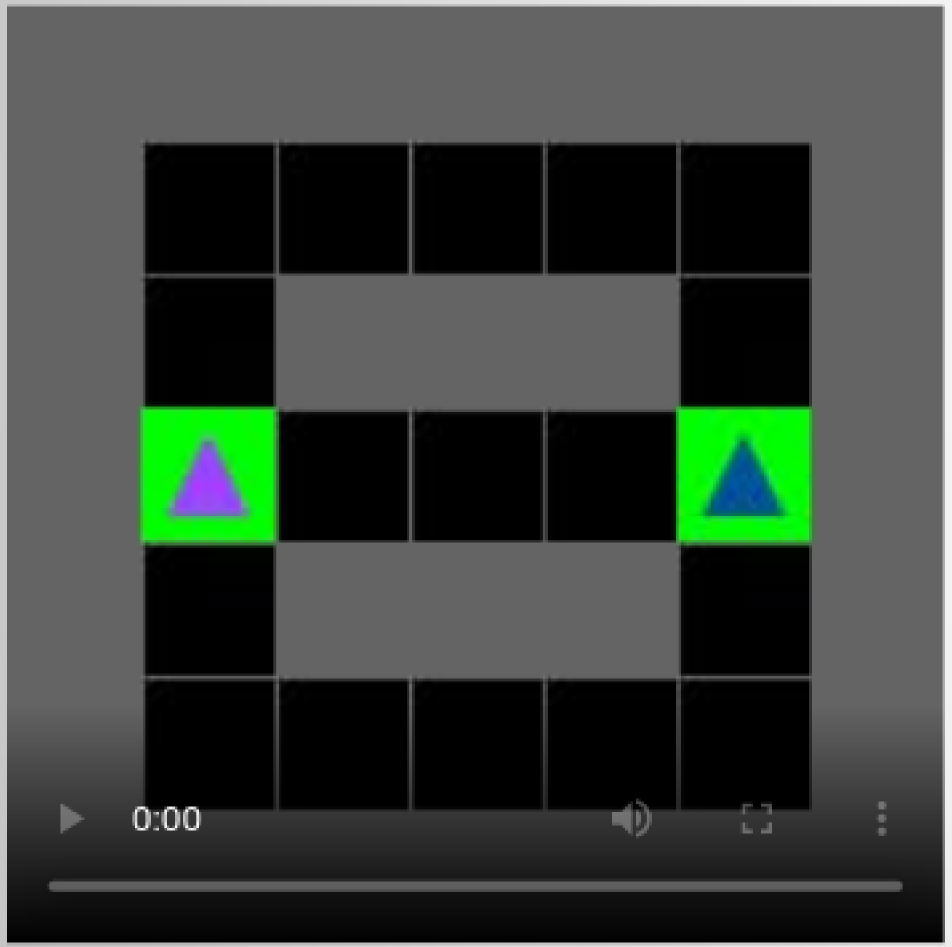


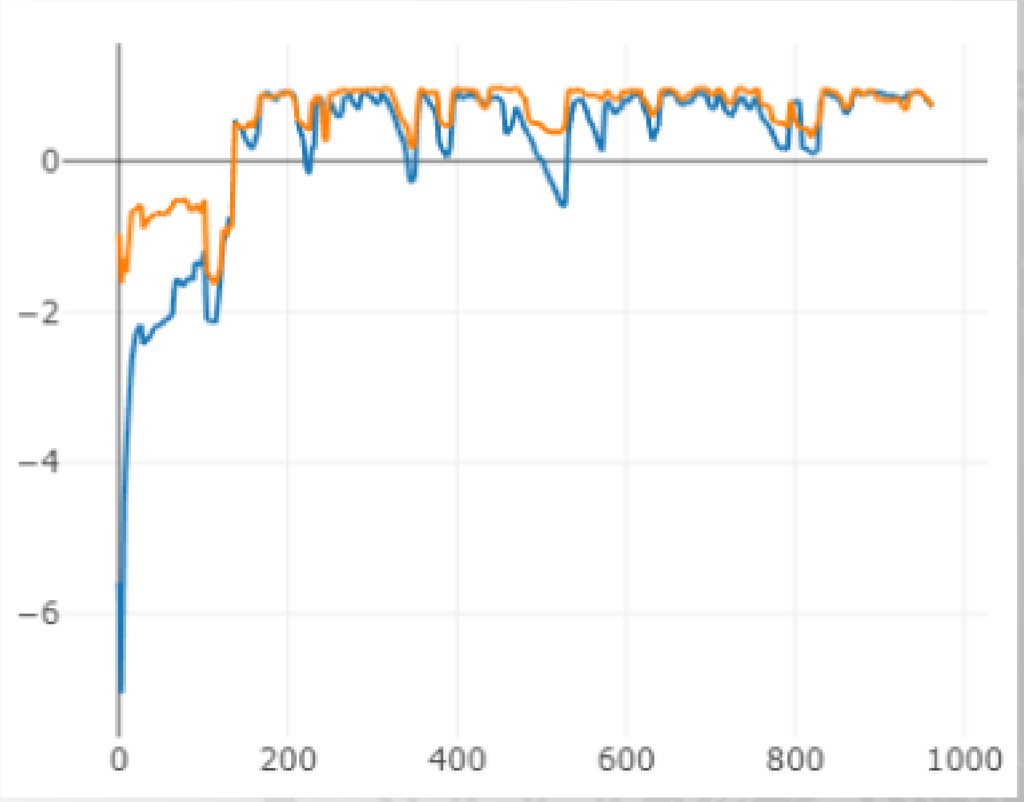


result #2 - DQN

-0.04 + 1 = 0.96

-0.07 + 1 = **0.93**





Challenges

TEAM:

- Remote team work
- Uncertain Uni schedule
- Issues: slow downs when waiting for reviews

TECH:

- Ray (Doc vs. source code)
- Different OSs
- Complexity of project/system
- Training multiple agents

Lessons Learned

- Avoid tests with long runtime or large memory footprint
- Pair sessions foster team work
- Iteratively adjust dev processes
- Explore/evaluate system-wide options as opposed to rigid assignment

Special Thanks: Sabrina Friedl

Possible extensions

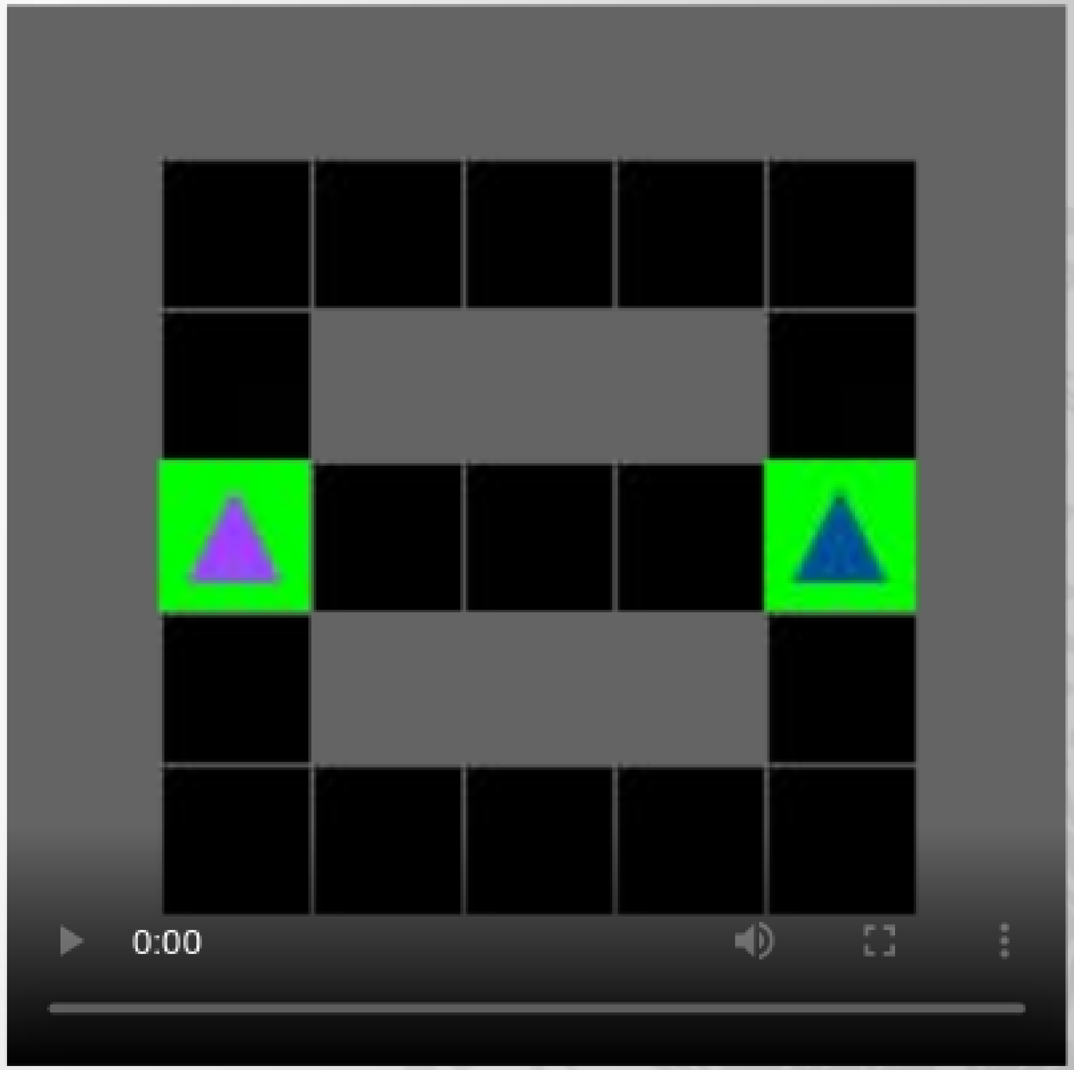
- QMix
- experiment with larger env
- more agents
- pick up item before goal (shopping)
- maze env

Questions?

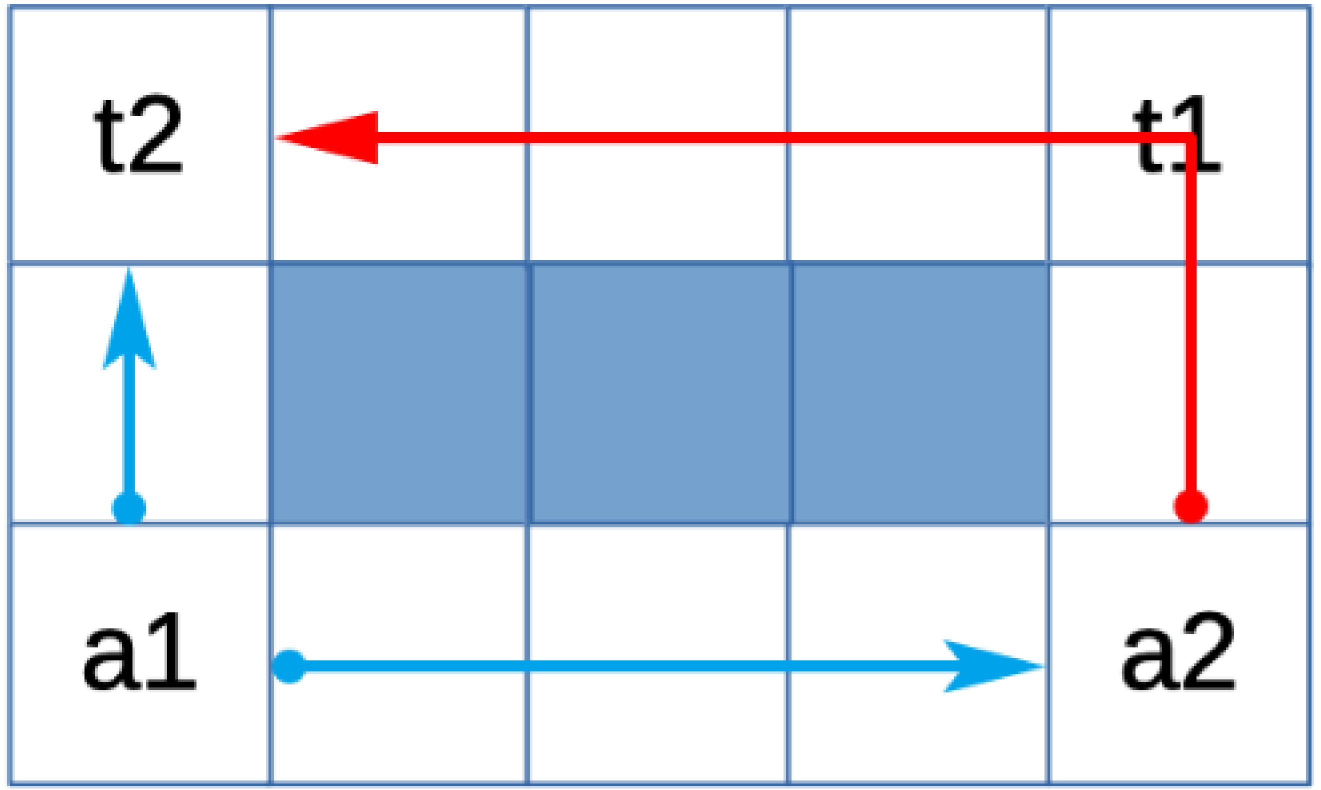
EXTRA

Independent 2x DQN

- Standing still
  - Min reward: -1.92
- Independent policies don't find a compromise
- Contrast to shared policy DQN w/ 2 agents

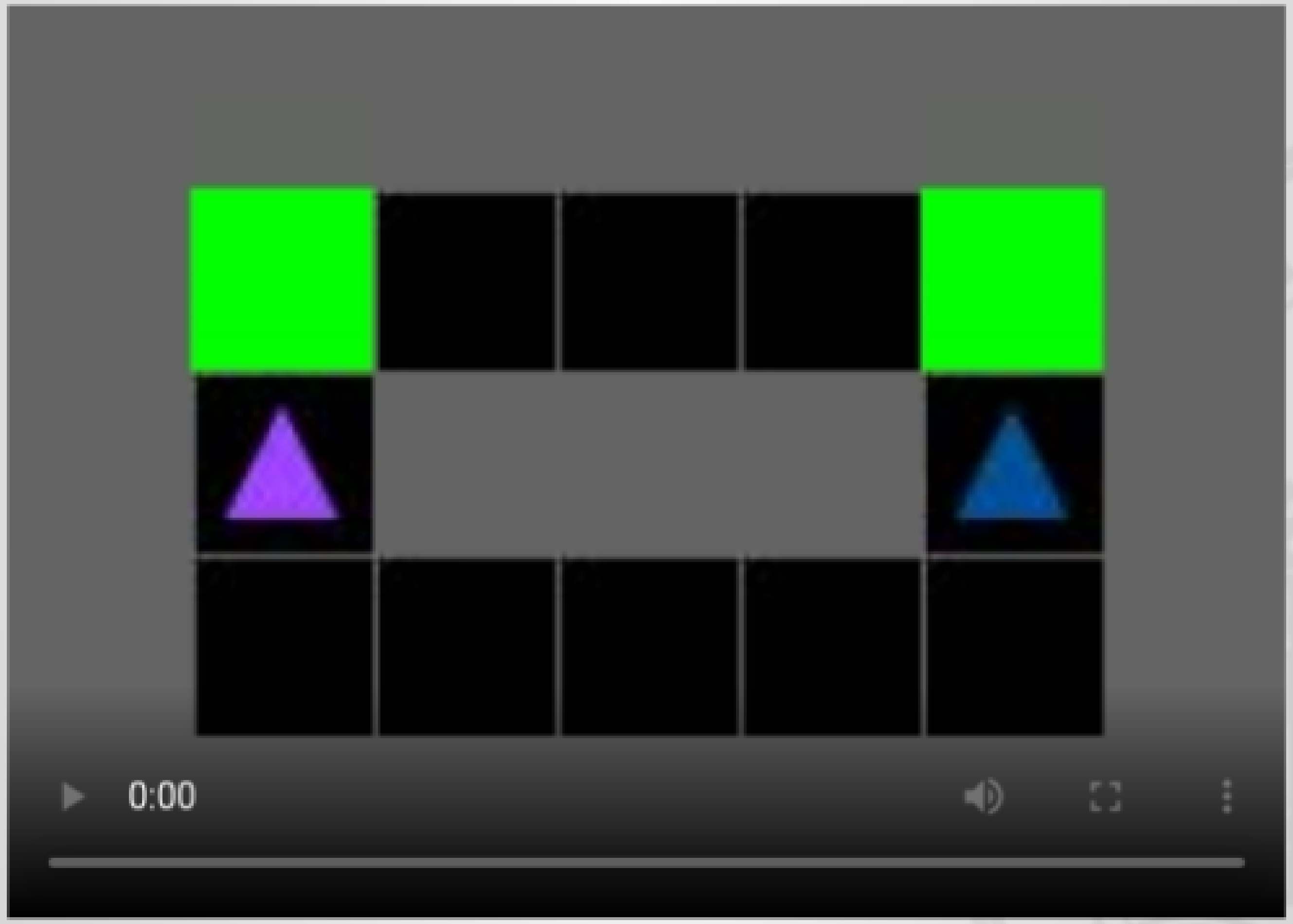


env #3

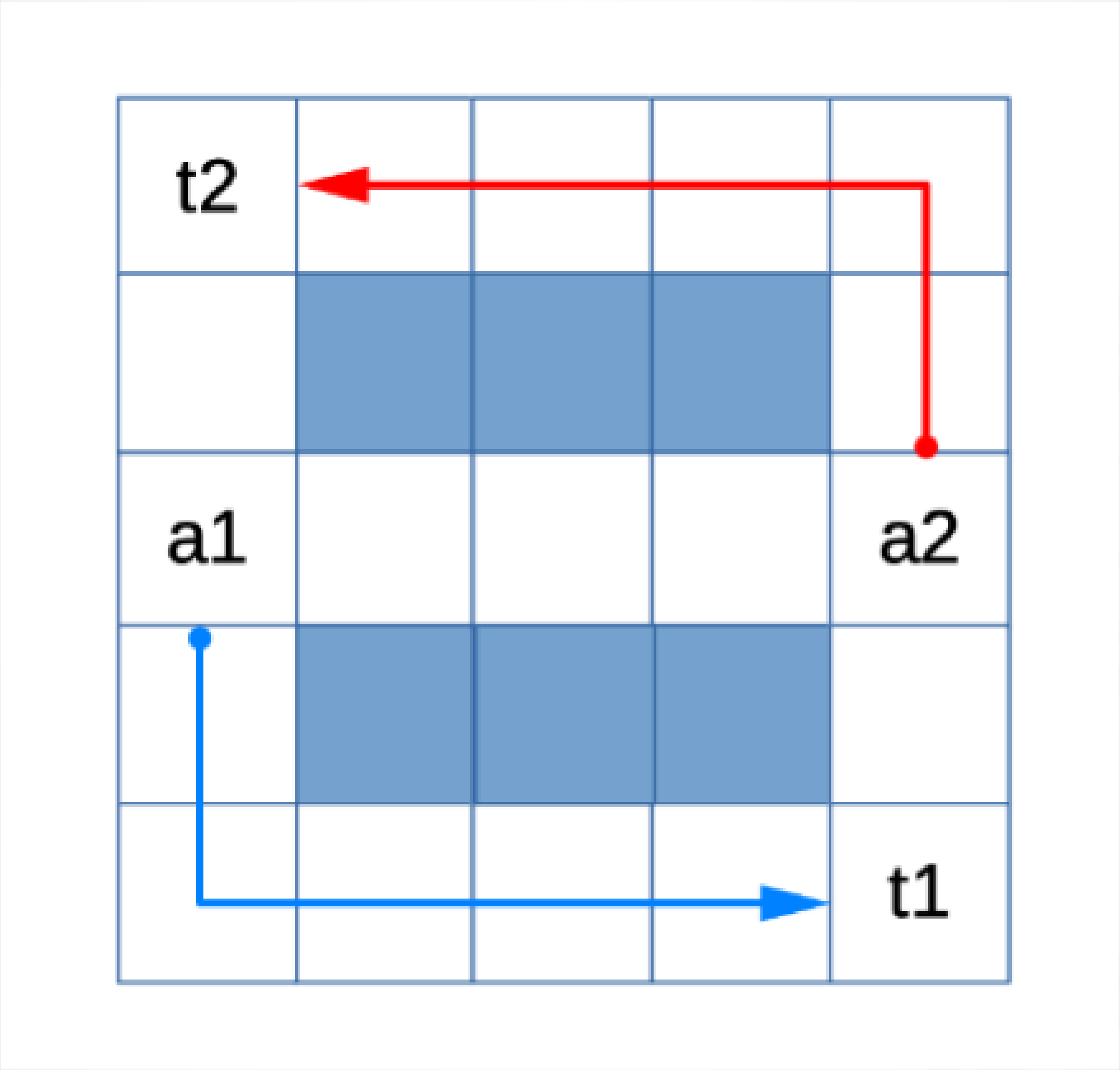


result #3 - dijkstra

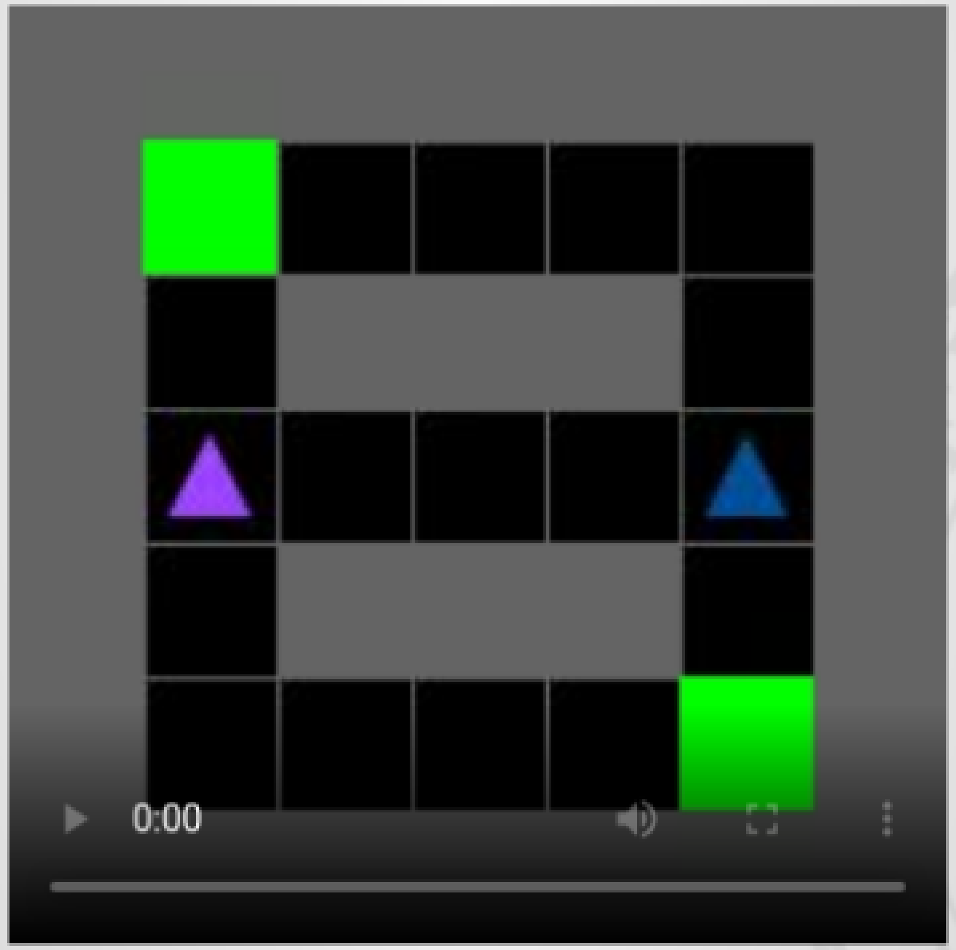
result #3 – dijkstra + DQN



env #4



result #4 - dijkstra



result #4 – dijkstra + DQN