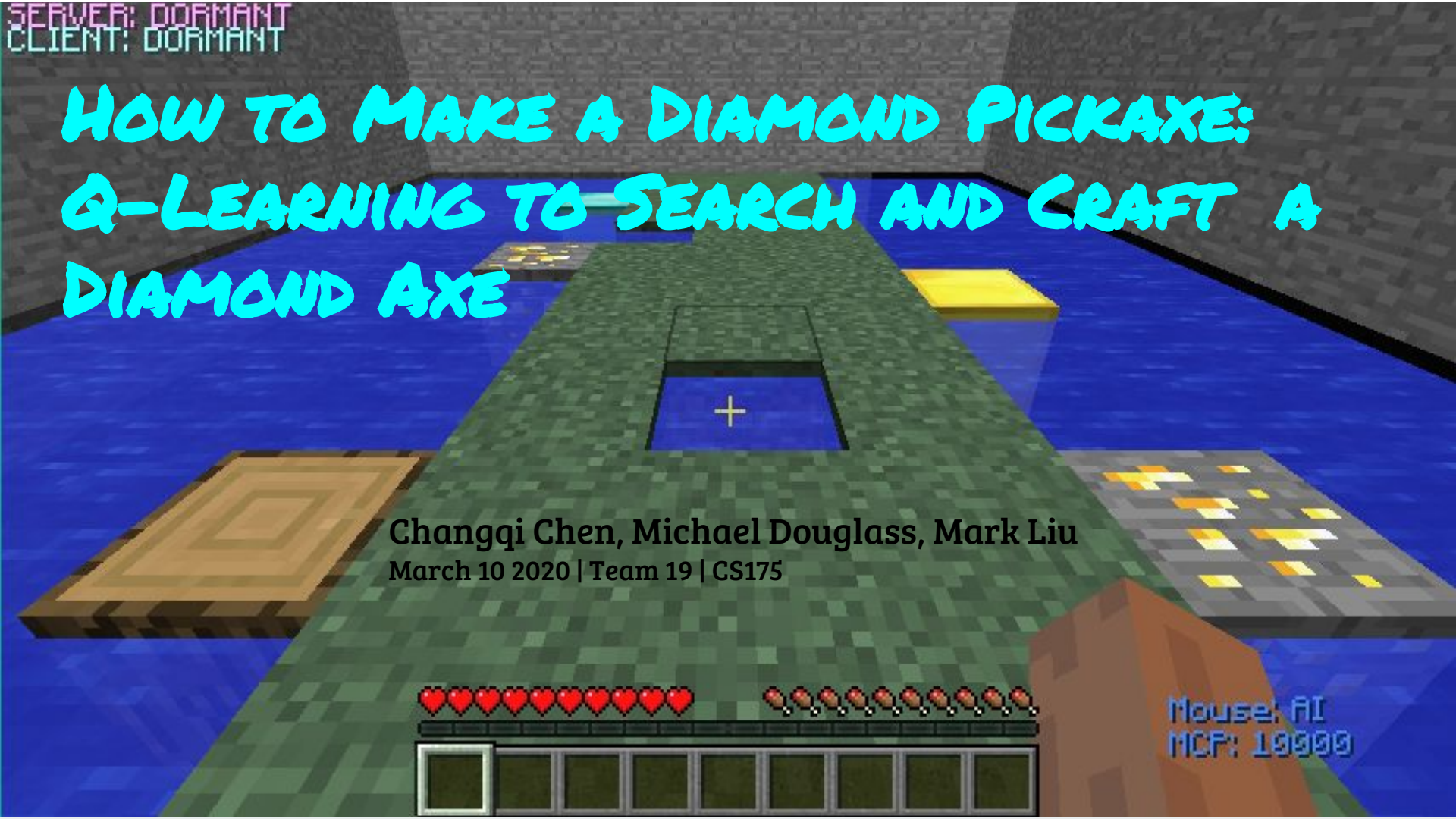


How to Make a Diamond Pickaxe: Q-Learning to Search and Craft a Diamond Axe

Changqi Chen, Michael Douglass, Mark Liu

March 10 2020 | Team 19 | CS175

Introduction

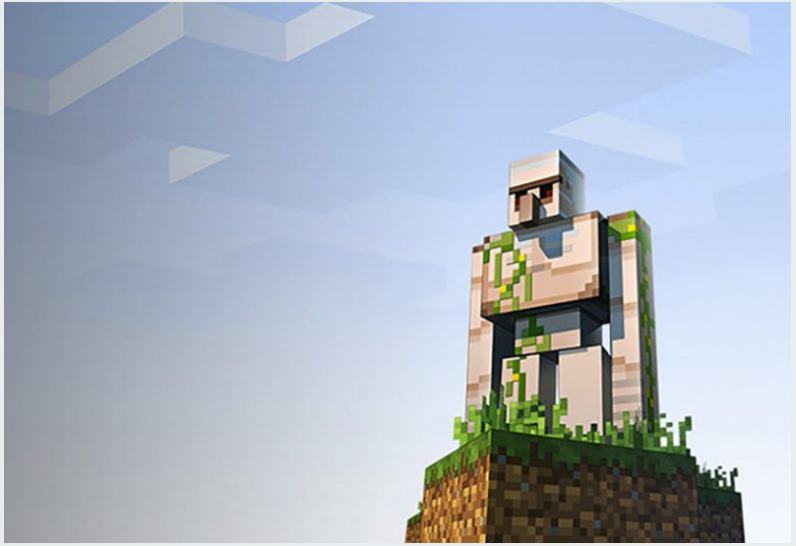
Introduction

- Malmo Platform for Minecraft.
- Our AI is aiming to imitate a task (such as mining) like a normal player. Search → Craft
- The AI workload procedure: Find wood → make wood pickaxe → use wood pickaxe to mine stone → Craft gold pickaxe → … → ultimately crate diamond pickaxe
- The program is using Q-Learning algorithm in searching the materials, learning how to make a specific item and getting more intelligent after each round.
- The experiments contains 1000 rounds with a 30s cap, a Q-table, and the environment is containing stone wall as boundary, water as trap, wood, gold, diamond blocks as the goto objects, and an algorithm to randomize the movement of the AI to imitate the randomization of real environment.

Background

Background

- Main Platforms are Malmo and Minecraft
- Normal Python Libraries: Json, sys, os, time, logging, random, MalmoPython
- Draw out the states with Library TKinter.
- Functions declarations are mainly from Python_Examples.
    - DrawQ, run, act, UpdateQTable, UpdateQTableTerminatingState
    - Various inventory checking functions
    - What's inside tho is heavily modified by us
- Structure of Agent, i.e Agent being a class, was also from Python_Examples.



Background

- Tabular Q-Learning Algorithm
  - Two action sets based on Map position
  - 51 regular map positions and 6 special map positions
  - Use the Bellman Equation in updating our policy.
- XML commands from various declarations and definitions from XML Schema Documentation.
- Mainly looked at existing examples from Python_Exmaples and adapted what we needed to suit our needs.

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

Bellman Equation

Problem Statement

Problem Statements

- The size of the map influence the performance.
- In earlier version, the AI will dig once it takes an action, but the AI tends to keep digging down or turning around without limits. Also, the z-axis in the project makes the whole learning complex and inefficient. Tried to add jumping move and a lava floor to prevent the agent from digging too much.
- The agent in malmo platform would dig everything even the bedrock, so it is hard to emulate the real minecraft.
- Confused about whether we should implement crafting before we fix the “dig through everything” problem.
- With the constant digging, it not only makes the environment hard to control, also the learning becomes inefficient which the agent will sometimes spend 100 rounds just to finish the task once.

Milestones

Milestone 1

1. Find/Acquire 2 wood logs. This will be enough to craft a wooden pickaxe.
2. Craft a wooden pickaxe
    - Requires crafting substeps
    - First crafting at least three planks
    - Then crafting at least two sticks from the planks
    - Finally combining the two previous steps to make a wooden pickaxe

- Challenges:
    - Digging Everything
    - One round crafting or piece-by-piece crafting

- Results:
    - Wooden Pickaxe!

Milestone 2

1. Find gold block and acquire it. This gives you 9 ingot, enough for a golden pickaxe.
    - Must have the wood blocks dug to acquire gold block.
2. Achieve a gold Pickaxe
    - Must have already crafted a wooden pickaxe
    - Using 3 gold ingot from the block and 2 sticks, create a gold pickaxe.

- Challenges:
    - One round crafting vs piece-by-piece crafting

- Result:
    - Golden Pickaxe!

Milestone 3

1. Find and Acquire a Diamond Block
    - Must have the gold block dug to acquire diamond block.
2. Craft a Diamond Pickaxe
    - Must have both wooden pickaxe and gold pickaxe
    - Using 3 diamond from the block and 2 sticks craft a pickaxe

- Challenges:
    - One round crafting vs piece-by-piece crafting

- Result:
    - Diamond Pickaxe!

Methods

1. The model class we use in our project to find the optimal policy is the tabular Q-learning.
2. It takes action A on state S, notes the reward and the next state S', then it chooses the max Q-Value in state S' then use all these info to update Q(S, A), then moves to S' and executes epsilon greedy action which does not necessarily result in taking action that has the max Q-Value in state S'

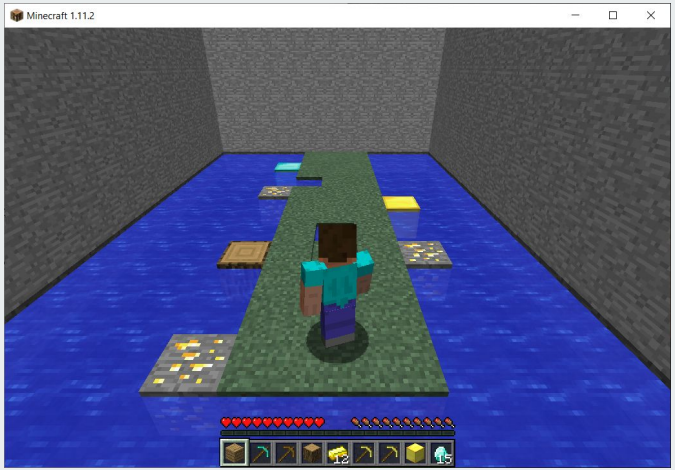
$$Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \max_a Q(S',a) - Q(S,A)\right]$$

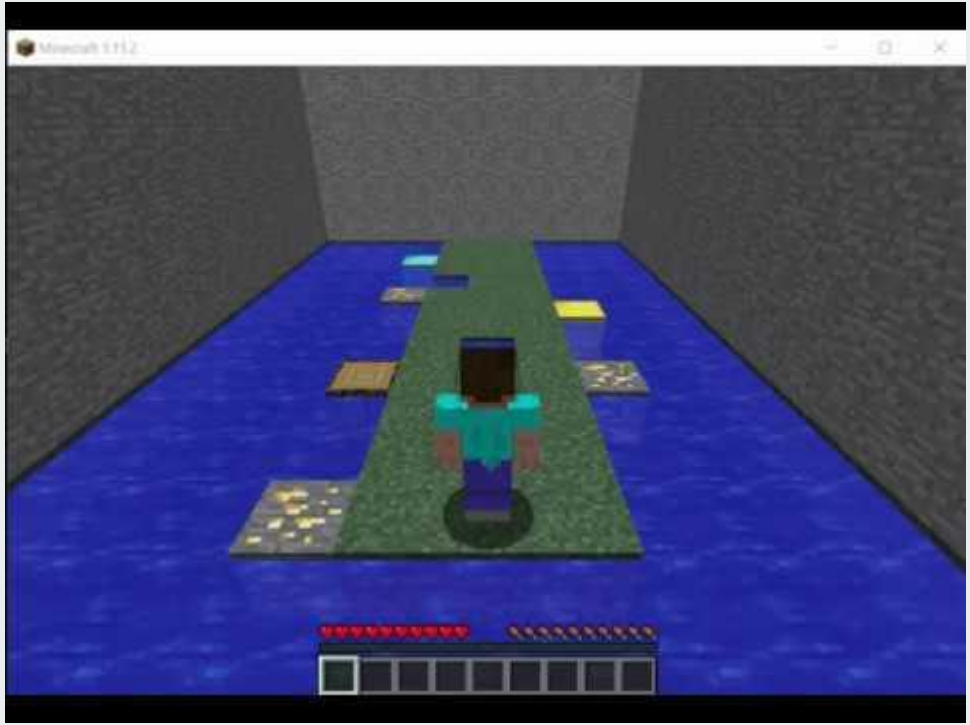
Experiments

- We try to increase the size of our map to make it more complex, but the large size of map make the AI acts very slowly. So we instead add some hole on our map to increase the complexity.
- Finally, we give up the idea to let the AI dig down, turn and jump since it will make it too complex and inefficient. Sometimes it will keep turn around without any other action.
- Since the agent can dig anything. We add some gold_ore block on our map. We only allow our agent to have the dig option when it stand on the gold_ore block. So it will prevent the agent from keep digging down.
- After fix the keep digging problem, we implement the crafting part for our agent.
- In previous version of our project, we run our agent with different learning rate to see which learning rate can make our agent perform best. We found out that the answer is 0.5.

Results

Currently, our agent will try to first dig enough log to make the sticks and the wooden pickaxe. Then the gold to make the golden pickaxe and finally the diamond pickaxe. The difficulty increases since the diamond block is most far from where agent start and there are traps on its way. It will learn to find the best way to reach the golden_ore block.





References

Sutton,R.S.andBarto,A.G.(2018).Reinforcementlearning:Anintroduction.MITpress.