CS551K Software Agents and Multi-Agent Systems

Firefly Agents Assessment 3 Report

Group members:

Zixiang Tang 52319604

Xiaocheng Ma 52319602

Zhichen Zhao 52319112

Xiangming Wang 52319606

Fan Yang 52320772

Implementation Overview

The basic goal of the game is to carry the designated block to the designated location according to the provided task completion instructions. The simplified strategy selection only completes the task that requires carrying one block, avoiding the complexity of connection operations.

Our agents design adopts a series of condition-based judgment logic, similar to a huge switch-case structure, to trigger corresponding action plans based on the agent's current state and environment.

Main Functions and Strategies

Overview

In essence, there are only two types of tasks based on the reward structure. One type requires two blocks and offers a reward of 40, while the other type only requires carrying one block and offers a reward of 10.

For simplicity, we choose to focus on the latter task type (which can still yield a high score). This means that each agent will follow the same script, independently carrying out its own work by transporting one item at a time (therefore eliminating the need for connect operations). Each robot will randomly move within its field of view until it discovers a Dispenser, at which point it will approach and pick up the item. It will then rotate to the appropriate direction and continue moving until it finds a Goal. Once a Goal is reached, the robot will submit the task with the earliest deadline that matches the condition (either b0 or b1), and resume random movement.

By adopting this strategy, each robot operates independently, focusing on efficiently completing the simpler tasks while still achieving competitive scores.

Step. “Random Movement and Direction Selection”:

When the agent is not performing a specific task, it will randomly select a direction to move. This is achieved through a calculation rule based on random numbers and a list of directions to ensure exploratory movement.

Step. “Handling Dispensers in Sight, and Request & Attach”:

When the agent needs a block and spots a Dispenser within its field of view, it will perform the action of taking the block. If the Dispenser is within one step, it will request it directly; if it's beyond one step, it will move towards the Dispenser.

Step. “Finding and Moving Towards the Goal”:

If the agent is carrying a block but is not at the target location, it will move based on the relative position of the goal to approach it in the shortest path.

Step. “Rotating and Submitting Tasks”:

When the agent is carrying a block and is at the target location, it will rotate appropriately based on its position and the relative direction of the target to ensure correct placement of the block.

Step. “Restore”: back to step 1, go for next task.

During random movement, if an agent moves one step at a time and selects opposite directions consecutively, it may result in the agent repeatedly stepping in place, particularly in maps where dispensers are unevenly distributed and the agent spawns in large empty areas devoid of dispensers. To mitigate this inefficiency, it is advisable to implement a strategy where the agent takes five steps at once, swiftly exiting the vacant zones. Given the dimensions of the map, a movement of five steps is deemed sufficient to navigate through and optimize the agent's mobility, ensuring a more effective traversal of the environment. This approach not only enhances the agent's ability to explore the terrain but also reduces the likelihood of stagnation in sparsely populated areas, thereby improving the overall efficiency of the task completion process.

Despite the incorporation of multifaceted scenarios into the algorithmic design, agents may still encounter operational impasses that hinder their ability to function effectively. Such predicaments can arise from being entirely encircled by obstacles, getting immobilized during rotational maneuvers, or experiencing contention among multiple agents over the acquisition of the same block. To address these challenges with a straightforward solution, the introduction of a stochastic mobility mechanism is proposed. This mechanism mandates the execution of random movements at predetermined intervals, irrespective of the agent's current phase of operation. This strategy aims to systematically disrupt any potential deadlock or stagnation scenarios, thereby facilitating the reinitiation of productive workflows. By integrating this random movement protocol, it becomes feasible to mitigate the issues associated with operational impasses, ensuring that agents can navigate out of immobilizing situations and resume their task-oriented activities without undue delay. This approach not only enhances the resilience of the agents against environmental constraints and competitive interactions but also contributes to the overall robustness and efficiency of the multi-agent system.

Results

Overall, the effect is good. The agent can find the dispenser quickly and submit the block. Most of the time, it can get a high score of 300 to 400, and even up to 600 if lucky. However, there may be situations where the agent is blocked or interferes with each other, resulting in a lower score. An agent that works normally can average 120~150 points.

In this multi-agent system, each robot (agent) is equipped with a vision range of five units, operating within a confined space of a 32x32 grid map, which features a relatively sparse distribution of obstacles. This architectural design ensures that, provided the dispensers are not positioned in an excessively skewed manner, agents can efficiently identify viable pathways, approximately ten squares in length, for task execution. Such a setup significantly minimizes the inefficiency typically associated with aimless, random movements, as the agents are able to navigate through the environment with a higher degree of purpose and directionality.

Moreover, the dynamics of task generation within this system are notably accelerated, surpassing the operational pace of the agents. This discrepancy between the rate of task availability and agent productivity negates the necessity for maintaining a comprehensive task list. Consequently, agents are afforded the flexibility to select and submit any block without the risk of contributing to inefficiencies or redundancies. This feature streamlines the operational workflow, enabling a seamless integration of task selection and completion processes. It effectively harnesses the agents' capacity for task execution without the burden of meticulous task allocation or prioritization, thereby optimizing the overall throughput of the system.

Possible Improvement

The code also has some areas that could be optimized in the future.

Optimized Task Selection

Currently, agents submit tasks without considering the deadline. Introducing a deadline-based priority selection logic would improve efficiency and increase the score. Agents should prioritize tasks with the earliest deadlines to maximize the number of completed tasks.

Improved pathfinding logic

After finding the target to go to, the agent will only move straightly towards the target. If blocked by multiple connected obstacles, it will get stuck. When the agent needs to rotate to the correct direction after obtaining the block, blocking may also occur, leading to submission failure.

Enhanced Agent Collaboration

Enhanced Collaboration Mechanism: Even when agents are only performing tasks with a score of 10, there is still a need for collaboration. Different agents may occupy the same block or block each other, which is one of the most important reasons for low scores. Therefore, it is necessary to introduce communication and information exchange among agents to avoid competition and optimize the overall performance. This could include mechanisms for task allocation, conflict resolution, and shared awareness of the environment and other agents' positions and actions. By Collaboration, agents can work together more effectively to complete tasks and achieve higher scores.

Memorization of path

Due to the significantly higher number of task generations compared to the agent's operational capacity, it is not necessary to focus on the specific tasks themselves. Instead, agents can simply submit any type of block continuously. If agents memorize the coordinates based on their starting position and repeatedly traverse the same route after completing any task, efficiency will be much higher. This approach leverages the agent's ability to learn and remember efficient paths, optimizing its performance and maximizing the number of blocks submitted. By using memorization, agents can avoid unnecessary exploration and directly focus on submitting blocks, thus increasing their overall productivity.