Iran University of Science & Technology

School of Computer Engineering

Assignment #3

Multi-agent system

BY:

DR. Naser Mozayani, Fall 2024

Teaching Assistants:

Fateme Raisi amjad

Due: 1404/02/27

Contents

Notes:

1. Submit the answers in a complete PDF file and the code for the questions in the .ipynb format (including the notebook cell outputs) in a compressed file named HW3_StudentID.zip by the specified deadline.
2. A total of 72+24 hours of delay in submitting the answers is allowed across all projects. After that, for each additional day of delay, 10% of the score will be deducted.
3. If a student submits the project earlier than the deadline and achieves 75% of the score, up to 24 hours will be added to their allowable delay time.
4. The maximum delay for submitting each assignment is 5 days, and after 5 days, submission will not be accepted.
5. The exercises must be performed individually, and group participation is not allowed.
6. It is important to note that the explanation of the code and the obtained results must be included in the PDF file. Code without a report will result in a score deduction.
7. The evaluation of the assignment will be based on the correctness of the solution and the completeness and accuracy of the report.
8. Please allocate sufficient time for the assignment and avoid leaving it until the last days.
9. You can ask your questions in the relevant group.

good luck.

Designing a Cooperative Multi-Agent LLM System (City in Flames)

In this exercise, you will extend the environment from Exercise 2 by introducing a second intelligent agent. Both agents are powered by Large Language Models (LLMs) and must operate collaboratively in the same dynamic, partially observable city. Through real-time perception, information sharing, and coordinated decision-making, the agents are expected to explore the environment, extinguish fires, and rescue civilians. This task emphasizes multi-agent cooperation, shared knowledge, and decentralized planning using LLMs.

Scenario Description:

A 15×15 grid-based city is initialized with fires, buildings, emergency stations, and civilians. Two LLM-powered agents are deployed and must cooperate in the environment. At the beginning, only the positions of the emergency stations are known. The agents must explore the city, share observations, and coordinate their actions through structured LLM prompts. The goal is to maximize the total score by collaboratively extinguishing fires and rescuing civilians within a limited number of time steps.

Environment Details and Game Rules:

Environment Components:

- Buildings: 7 immovable obstacles that block the agent’s movement. The agent cannot enter or pass through these cells.
- Emergency Stations: Located at the top-left corner ([0, 0]) and the bottom-right corner ([14, 14]). Rescued civilians must be delivered to these locations. Fire cannot spread into these cells.
- Civilians: 8 individuals randomly distributed across the city. The agent must rescue and transport them to an emergency station.
- Fires:

  - Type 1 Fires: 5 small fires randomly placed, each occupying one cell and independently extinguishable.
  - Type 2 fires: 3 large fires that are randomly located, each occupying one cell and requiring the cooperation of 2 agents simultaneously to extinguish.

- Agents**:** Two LLM-powered agents are initialized at random locations. Each can move independently in four directions: up, down, left, and right.

Movement and Interaction Rules:

- Each agent can move to any of the four adjacent cells (up, down, left, right) at each step.
- Agents can simultaneously occupy a cell or pass through each other.
- To extinguish a fire, the agent must enter a fire-containing cell and execute a special action. Simply entering the cell does not automatically extinguish the fire.
- To rescue a civilian, the agent must enter the civilian's cell, execute the rescue action, and then transport the civilian to an emergency station.
- Each agent can carry only one civilian at a time and should prioritize transporting them safely.

Fire Spread:

- Every 5 consecutive agent moves, the fires spreads randomly to an adjacent cell (right, left, up, down).
- Fire cannot spread to a cell containing an emergency station and buildings.
- If all surrounding cells of a fire are already burning, it will not spread further.
- The agent can pass through burning cells.
- If the agent is carrying a civilian, the civilian is not harmed while passing through fire.

Perception and Exploration:

- At the start, both agents have no knowledge of fires locations, buildings, or civilians.

- The environment is completely unknown, and the agents must explore to discover it.
- Agents only have knowledge of the emergency station's location.
- Each agent can perceive a 5×5 area around itself.
- Agents share their observations after each move to update a shared map of the environment.
- To fully uncover the map, agents must move and explore from different positions.
- If fire spreads beyond the agent’s sensor range(5×5 area), agents will not be aware of the change.

Scoring System:

- +10 points for extinguishing each Type 1 Fire.
- +25 points for extinguishing each Type 2 Fire.
- +50 points for rescuing and delivering a civilian to an emergency station.
- -100 points if a civilian dies due to fire spread.

Game End Conditions:

- The game ends after 50 time steps.
- If all fires are extinguished and all civilians are rescued within this limit, the game is successfully completed.
- If fires remain or not all civilians are rescued, the agents fail.

Additional Notes:

- All components (fires, buildings, civilians, emergency stations) are randomly and unpredictably placed.
- To extinguish fires or rescue civilians, agents must perform specific actions—simply entering a cell is not sufficient.
- To extinguish Type 2 fires, two agents must cooperate and extinguish the fire together and simultaneously with the extinguishing action.

- Each agent must explore the environment through limited perception and continuously share observations to maintain a common knowledge map.
- The agent must explore and observe the environment to gather information for decision-making.
- Agents coordinate actions using structured prompts generated for LLM reasoning.
- Use the Python programming language.
- Display the city map and the map discovered by the agent from its surroundings, and print them on the .ipynb output.

Key Enhancements from Exercise 1 and 2:

- Replaces single-agent execution with a cooperative multi-agent system powered by LLMs.
- Cooperation between agents.
- Adds real-time information sharing between agents to build a shared map of the environment.
- Enables joint planning and task division through structured prompts, where each agent considers the other's position and objective before acting.
- Maintains fire spread rules and game dynamics from Exercise 1.
- Enhances transparency and decision quality by requiring each agent to explain its actions in context of the team strategy.

Agent Design Requirements:

I. Perception and Memory

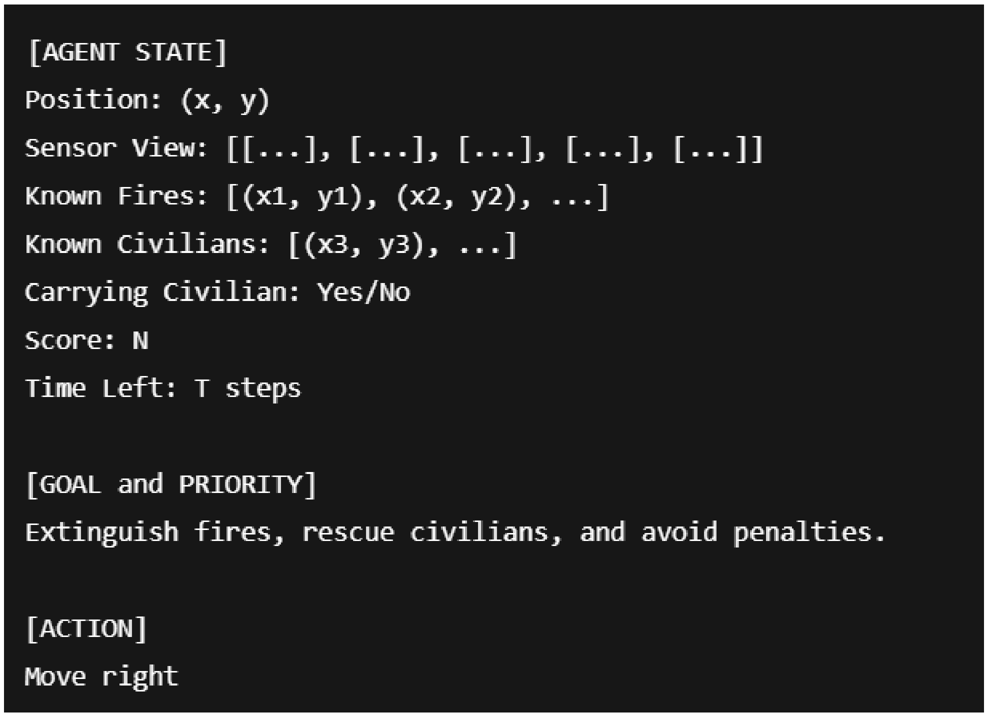
- Each agent maintains a personal view of its 5×5 surroundings.
- After each move, both agents exchange their perceptions to update a shared 15×15 map of the environment.
- Log discovered locations of:

    - Fires
    - Civilians
    - Buildings
- Monitor:
    - Whether the agent is carrying a civilian
    - Nearest emergency station
- LLM-Driven Coordinated Decision-Making
- At each step, both agents generate a structured prompt for the LLM that includes:
    - Agent’s current coordinates and 5×5 perception view
    - Known map entities (fires, civilians, buildings)
    - Current score, remaining time steps
    - Carrying status of both agents (whether a civilian is onboard)
    - Last known position of the other agent
    - …
- The LLM should return one valid action per agent:
    - Move up/down/left/right
    - Extinguish fire
    - Rescue civilian
    - Deliver civilian
- Invalid actions result in either no action or a fallback random valid action.

LLM Integration Guidelines:

Example Prompt Format :

This prompt format must be generated separately for both agents, based on their individual perceptions and shared knowledge.



```
[AGENT STATE]
Position: (x, y)
Sensor View: [[...], [...], [...], [...], [...]]
Known Fires: [(x1, y1), (x2, y2), ...]
Known Civilians: [(x3, y3), ...]
Carrying Civilian: Yes/No
Score: N
Time Left: T steps

[GOAL and PRIORITY]
Extinguish fires, rescue civilians, and avoid penalties.

[ACTION]
Move right
```

Response Expectations

- Validate the LLM output to ensure it maps to a valid action.
- Require the LLM to include an explanation of reasoning, such as:

  Agent 1 Action: Move down

  Reason: Moving down to rescue civilian located at (5,4), avoiding immediate fire threats and buildings.

  Agent 2 Action: Move right

  Reason: Moving right towards emergency station at (14,14) to safely deliver the carried civilian, while also avoiding nearby fire at (11,13).

Evaluation Criteria and System Constraints:

To better guide implementation and encourage thoughtful design, the following evaluation criteria and system constraints are in place:

Evaluation Criteria

Your solution will be assessed based on:

1. Communication Efficiency(Quality of the problem statement and request from LLM)
   - Based on successful extinguishing of fires and civilian rescues.
   - How effectively agents share necessary observations.
   - Are they over-communicating or repeating redundant data?
2. Exploration Coverage
   - How much of the environment is discovered over time?
   - Are both agents contributing meaningfully to exploration?
3. Task Coordination
   - Evidence of cooperation: e.g., synchronized actions for Type 2 fires.
   - Are agents avoiding duplicated efforts (e.g., rescuing the same civilian)?
4. Decision Validity
   - Ratio of valid vs. invalid actions generated by the LLM.
   - Are decisions backed by sound reasoning?
5. Comparison with Single-Agent Baseline
   - Clear performance improvement over the prior single-agent system.

System Constraints

Your implementation must take the following into account:

1. Communication Constraints
   - Agents can only share information once per move.

Implementation Requirements:

- Use the Python programming language.
- Use either open-source LLMs (e.g., models from Hugging Face) or free-tier APIs**. You can use the hugchat package from pypi.**
- Display the following in a Jupyter Notebook (.ipynb):
  - Full city map (ground truth)
  - Agent’s discovered map over time

- Provide a performance report comparing:
  - Cooperative multi-agent LLM system
  - Single-agent system (Exercise 2)

Objective of the Exercise:

To develop a multi-agent system where two LLM-driven agents collaborate by sharing information and making coordinated decisions in a partially observable, dynamic environment. It serves as a foundation for simulating more complex collaborative AI systems and real-world multi-agent reasoning tasks.