AI50 Project
Patrolling problem

- Coordinating multiple agents to navigate an environment represented as a graph
- Ensuring that all strategic zones or points are visited regularly
- Visiting strategic locations in a region at regular intervals.

Context: Military Base

1

Reinforced Security

The military base requires constant surveillance to ensure the safety of the site and personnel.

2

Complex environment

The base has many obstacles and risk areas, making the movement of agents more difficult.

3

Critical Optimization

Patrol routes must be carefully planned to effectively cover the entire area.

Constraints

In the context of patrolling a military base, agents must contend with various constraints that directly impact their ability to conduct effective and continuous patrols.

- **Limited number of agents**
- **Mandatory return to starting node**
- **Variable Weather Conditions**

Objective function

Our **objective function** is to minimize idleness across the graph G. The idleness of a node i at time t, denoted $I_i(t)$, is defined as the time elapsed since the last visit by an agent to that node.

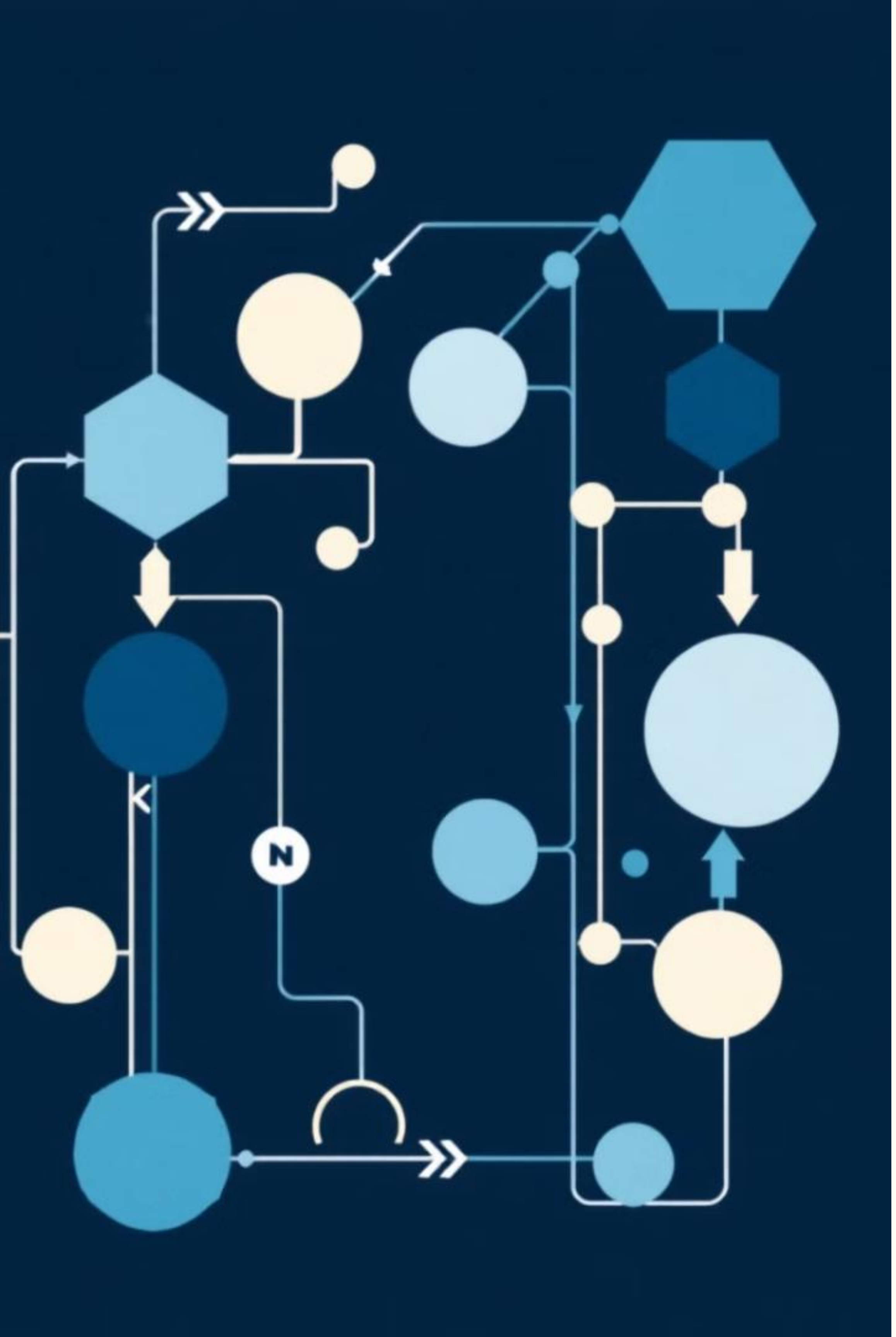
Two criterias :

Worst Idleness (WI) :

$$WI_t(G) = \max_{i=1,\ldots,N} I_i(t)$$

Average Idleness (AI) :

$$AI_t(G) = \frac{1}{N} \sum_{i=1}^{N} I_i(t)$$



Eulerian Algorithm

1

Definition

A Eulerian circuit is a path in a graph that traverses each edge exactly once before returning to the starting point.

2

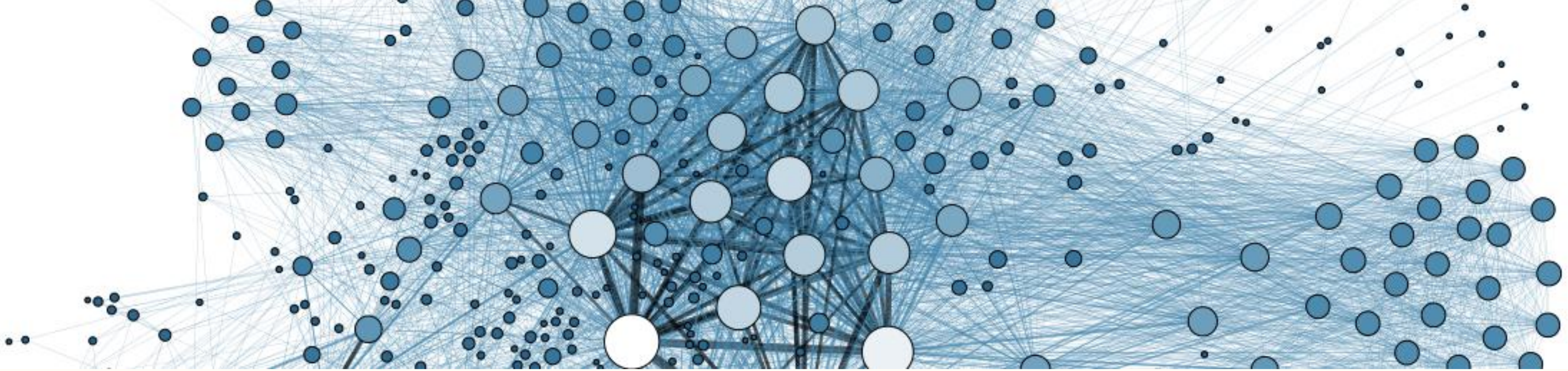
Advantages

The complexity is reduced once the graph is divided, as each agent follows a fixed route.

3

Disadvantages

Each agent follows a fixed path with no flexibility and without sharing information with other agents.



Starting nodes : 2 approaches

Single Initial Node

All agents begin their patrol from the same initial node.

Dispersed Deployment

In this approach, agents are spread out across the graph to occupy the nodes furthest from one another. This distribution phase is determined by an evolutionary algorithm, serving as a heuristic to form pre-cycles.

Reactive agents

Definition

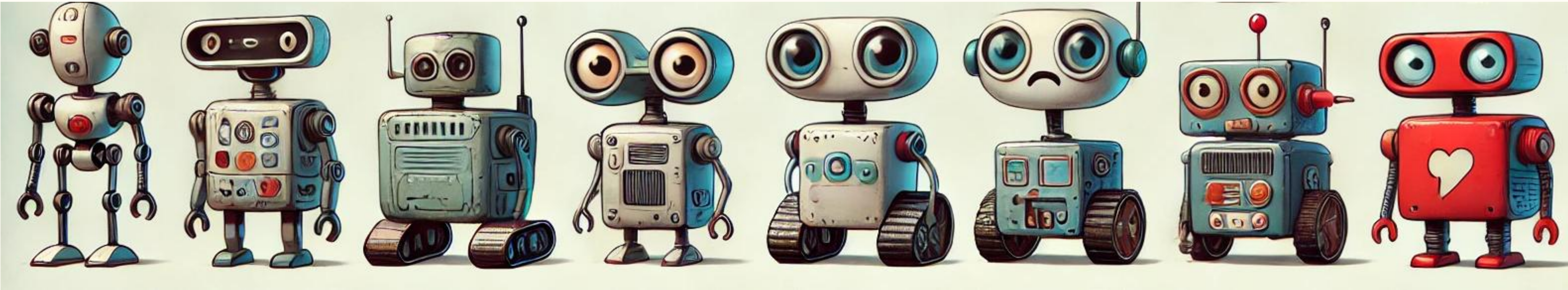
Each reactive agent moves towards the neighboring zone that has been neglected the longest (i.e the one with the highest idleness).

Advantages

This algorithm is simple to implement and does not require coordination between agents.

Disadvantages

This method is flexible and can quickly adjust to unforeseen events but may lack optimality and communication between agents.



Cognitive agents

1

Advantages

Enables a more strategic patrol by targeting high-idleness nodes across the entire graph and coordinating agent movements to improve coverage.

2

Definition

Cognitive agents can plan their movement by considering the entire territory. Each agent plans its route to the node with the highest idleness (even if that node is not a direct neighbor) by using a shortest-path algorithm, such as Dijkstra or A*.

3

Disadvantages

Higher computational complexity due to path planning and communication overhead.

ACO Algorithm

Definition

This approach is inspired by the behavior of ants, where agents (or "ants") deposit pheromones on paths they travel and prioritize paths that have not been visited recently.

Advantages

The ACO approach dynamically balances exploration and exploitation of paths, adapting over time to minimize idleness across the graph.

Disadvantages

Requires repeated iterations to converge to an optimal solution and may be computationally intensive, especially on larger graphs.

Hybrid Algorithm ACO/EA

Overview

The hybrid algorithm combines Ant Colony Optimization (ACO) and Evolutionary Algorithms (EA) to address the Multi-Agent Patrolling Problem (MAPP). The goal is to optimize patrolling strategies, ensuring that all critical zones are visited regularly while minimizing idle times.

Process

Step 1: Evolutionary Algorithm (EA)

- Objective: Find the optimal initial distribution of agents.
- Process:
  - Initialize population of solutions (agent distributions).
  - Evaluate solutions based on distance and coverage.
  - Select, crossover, and mutate solutions to generate better configurations.
  - Repeat for multiple generations until an optimal agent layout is achieved.

Step 2: Ant Colony Optimization (ACO)

Benefits

- Flexibility: Handles heterogeneous agents with varying speeds and priorities.
- Efficiency: optimal solution, balancing exploration and exploitation.
- Scalability: Can be adapted for larger, more complex environments with multiple agents.

Stack and tools

Python

Programming language used for implementing algorithms.

NetworkX

Python library for graph manipulation and analysis.

Visualisation

Simulation

•**Pygame**: To visualize agent movements within the environment and create graphical interfaces.

Conclusion and next steps

1

Development of the tools

Development of the vizualisation of the agents in the environment

2

Development of the different models

Setting up models and fine tuning hyperparameters

3

Compare / Ameliorate the models

Compare the models developed and try to reach the best performance