Route Optimization for Order Picking in Distribution Centers using Reinforcement Learning based Genetic Algorithm

By:
Matineh Rangzan

Supervisor:
Dr. Hossein Akbaripour

BSc Thesis in Industrial Engineering

Amirkabir University of Technology (Tehran Polytechnic)
Department of Industrial Engineering & Management Systems

Summer 2023

Introduction

The goal of this research is to use a **hybrid approach combining Multi-Agent Q-Learning (MAQL) and Genetic Algorithms** for **optimizing pathfinding** for Order Picking in distribution centers. This approach investigates the use of MAQL for finding the optimal path in the shortest time. By proposing a strategy for a centralized network of reinforcement learning agents that share information via a Genetic Algorithm and pass it to the next generation.

Contents

Order Picking in Distribution Centers

1

Order Picking

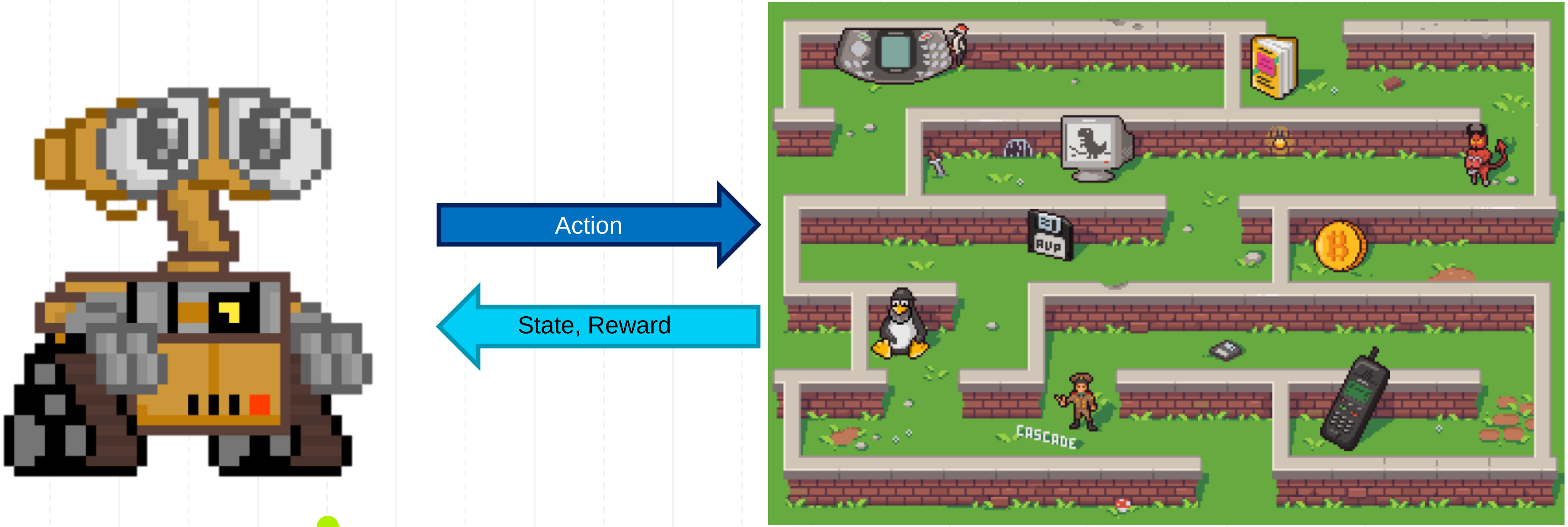
Among the various **warehousing processes** that have to be completed in a company, order picking, which is commonly defined as the process of retrieving items from their storage locations in response to customer orders, is considered one of the **most time-consuming and work-intensive** ones. It is estimated that it accounts for up to **55%** of the total warehouse operating costs.

2 Reinforcement Learning

Q-Learning

Q-Learning artificial intelligence algorithm is a reinforcement learning algorithm. Q-Learning **does not require any supervision** and can check an environment by simply specifying several parameters and **hyperparameters** and the method that gives it the best scores. In this method, it is **only** necessary to define the **environment** and the algorithm will find the path by itself.

How does it work?



The Bellman Equation

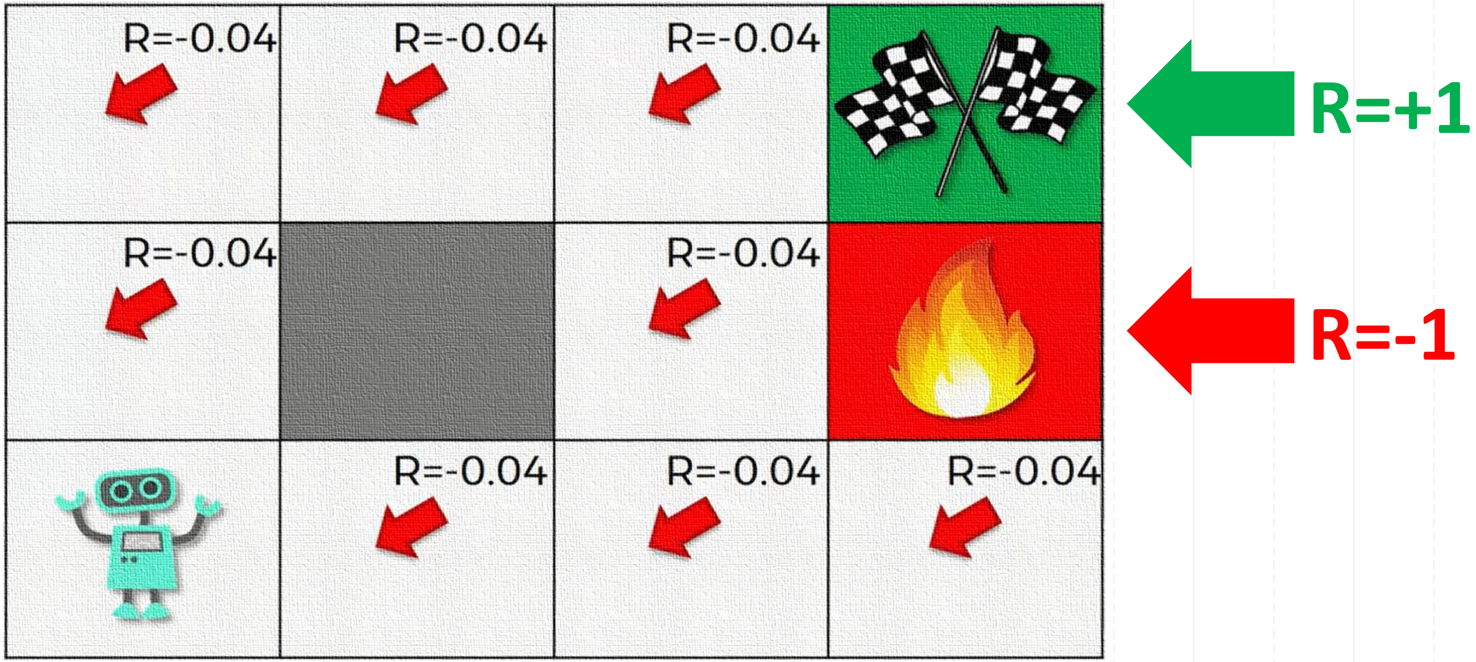
$$V(s) = \max_a (R(s,a) + \gamma V(s'))$$

V=0.81	V=0.9	V=1	
V=0.73		V=0.9	
V=0.66	V=0.73	V=0.81	V=0.73

Concepts:

- s – State
- a – Action
- R – Reward
- γ – Discount

Living Penalty (Rewards)



Temporal Difference

$$TD(s,a) = \overbrace{\boxed{R(s,a) + \gamma \max_{a'} Q(s',a')}}^{\text{now}} - \overbrace{\boxed{Q_{t-1}(s,a)}}^{\text{before}}$$

$$\boxed{Q_t(s,a) = Q_{t-1}(s,a) + \alpha TD_t(a,s)}$$

$$Q_t(s,a) = Q_{t-1}(s,a) + \alpha\left(R(s,a) + \gamma \max_{a'} Q(s',a') - Q_{t-1}(s,a)\right)$$

Concepts:

- s – State
- a - Action
- R – Reward
- γ – Discount
- α – Learning Rate

Algorithm

Q Learning

Initialize $\mathbf{Q}(\mathbf{s}, \mathbf{a})$ for all $\mathbf{s} \in \boldsymbol{\mathcal{S}}$ and $\mathbf{a} \in \boldsymbol{\mathcal{A}}(\mathbf{s})$

Loop forever:

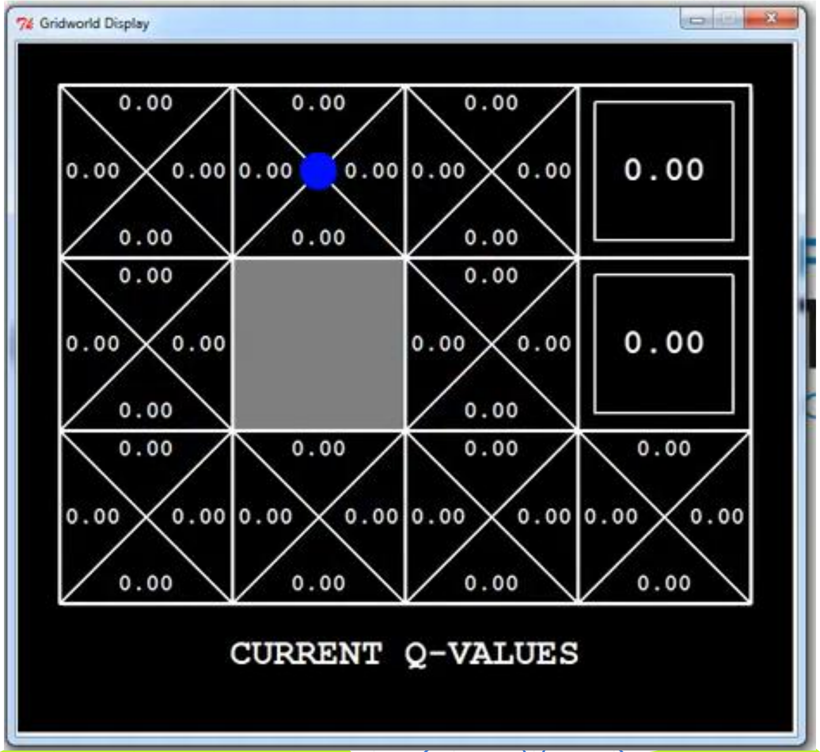
(a) $\boldsymbol{S} \leftarrow$ current (nonterminal) state

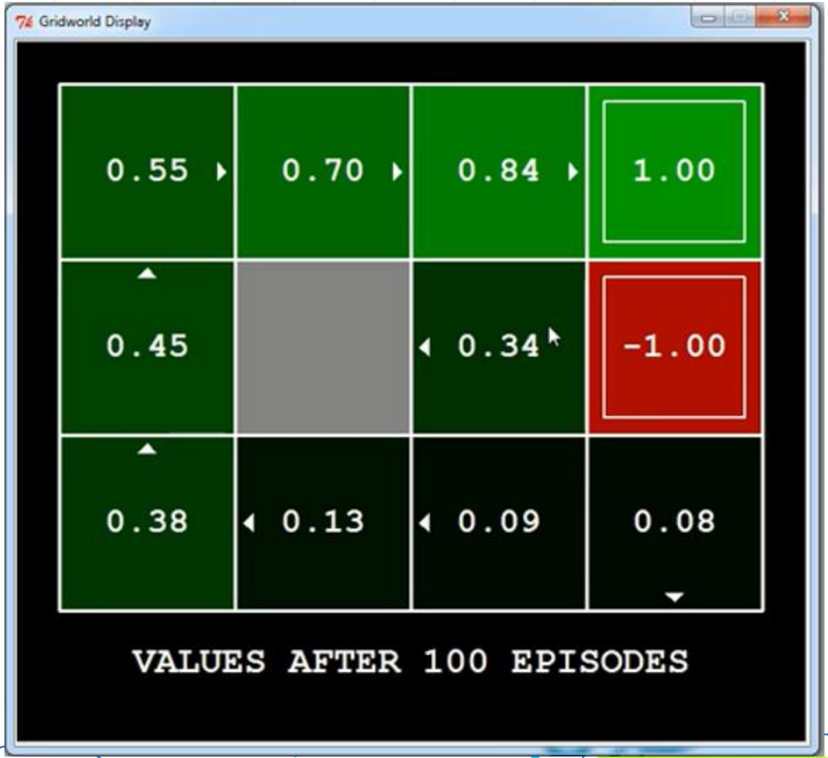
(b) $\boldsymbol{A} \leftarrow \boldsymbol{\varepsilon}$ -greedy $(\boldsymbol{S}, \boldsymbol{Q})$

(c) $\boldsymbol{Take\ action}$ A; $\boldsymbol{observe\ Resultant\ reward, R, and\ state, S'}$

(d) $\boldsymbol{Q(S, A) \leftarrow Q(S, A) + \alpha\left[R + \gamma \max_{a} Q(S', a) - Q(S, A)\right]}$

Training


Gridworld Display
CURRENT Q-VALUES


Gridworld Display
0.55
0.70
0.84
1.00
0.45
0.34
-1.00
0.38
0.13
0.09
0.08
VALUES AFTER 100 EPISODES

Multi-Agent Reinforcement Learning

3

Introduction

When we do multi-agents reinforcement learning (MARL), we are in a situation where we have multiple agents **that share and interact in a common environment.**

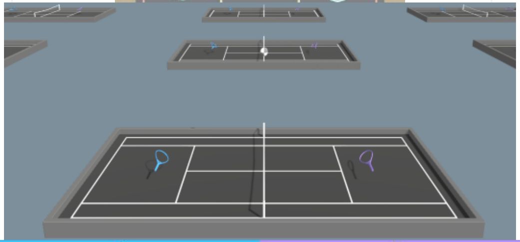
For instance, you can think of a warehouse where **multiple robots need to navigate to load and unload packages.**

Different types of multi-agent environments

Given that, in a multi-agent system, agents interact with other agents, we can have different types of environments:

1) **Cooperative environments**

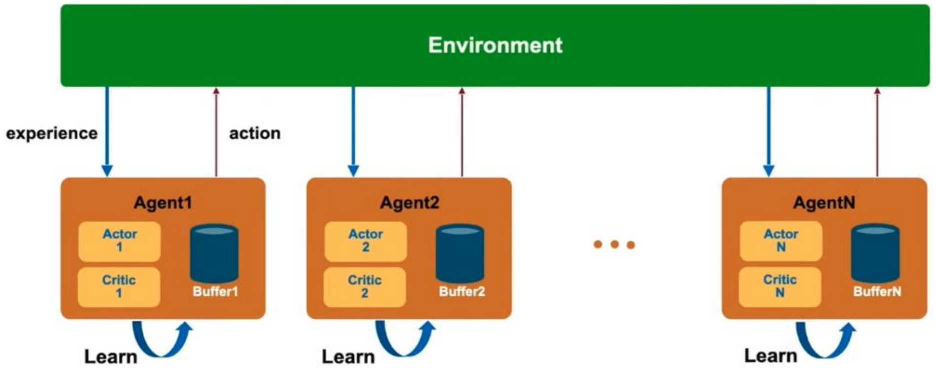
2) **Competitive/Adversarial environments**



3) **Mixed of both adversarial and cooperative**

MARL Approaches

Decentralized



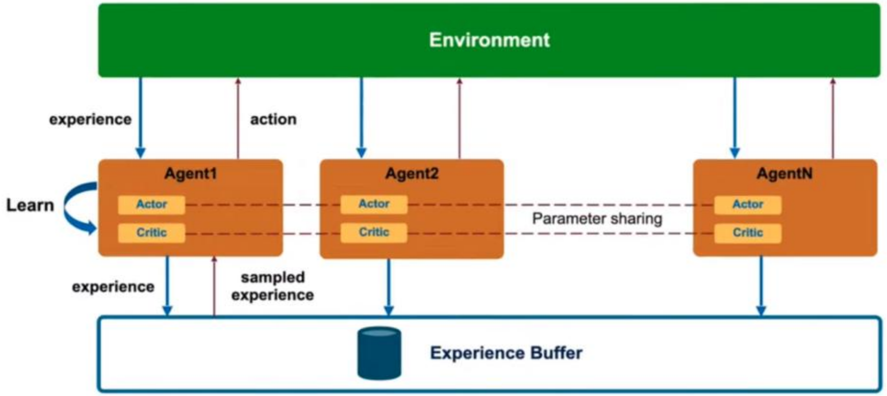
Benefits:

- Simplified system design

Drawbacks:

- Doesn't know state of other agents.
- Non-Stationary environment

Centralized



Benefits:

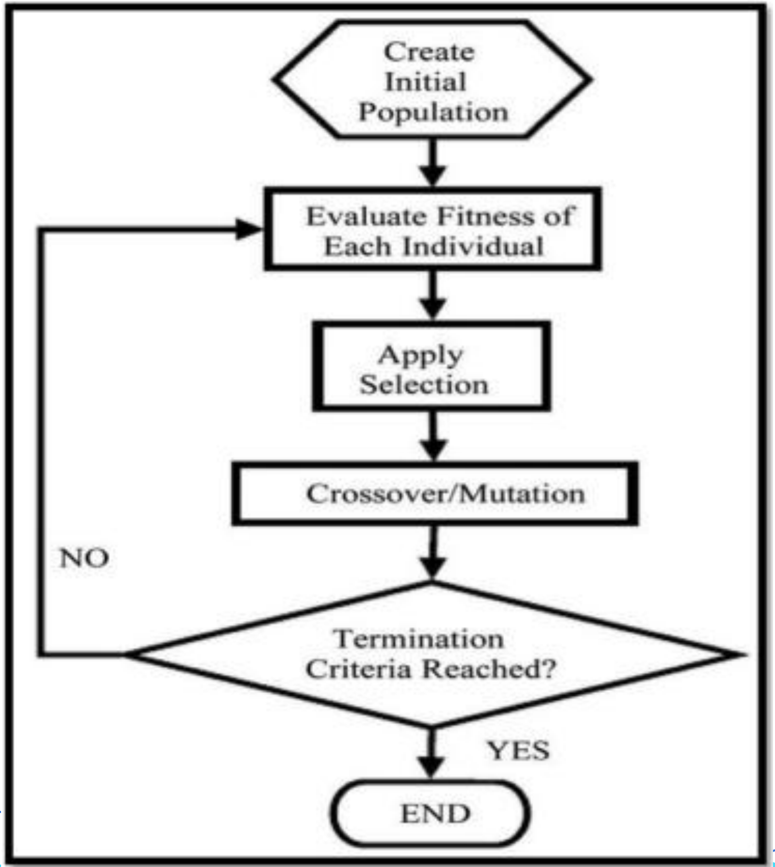
- Agents learn from collective experiences
- Stationary environment

Genetic Algorithm 4

INTRODUCTION

Genetic Algorithm (GA) is a **search-based optimization technique** based on the principles of Genetics and **Natural Selection**. It is frequently used to find **optimal or near-optimal** solutions to difficult problems which otherwise would take a lifetime to solve.

BASIC STRUCTURE OF GENETIC ALGORITHM



GENOTYPE REPRESENTATION (INFORMATION ENCODING)

Binary Representation

0	0	1	0	1	1	1	0	0	1

Real Valued Representation

0.5	0.2	0.6	0.8	0.8	0.4	0.3	0.2	0.1	0.9

Integer Representation

1	2	3	4	3	2	4	1	2	1

Permutation Representation

E1	E5	E9	E8	E7	E4	E2	E3	E6	E0

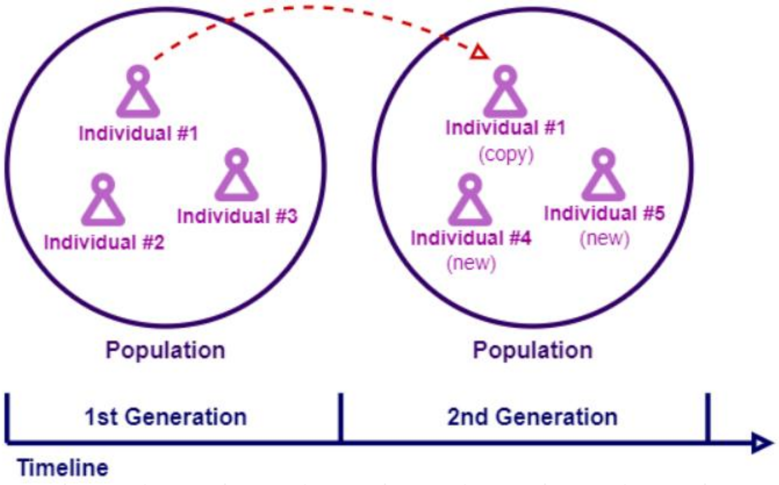
GENETIC OPERATORS

Three major operations of genetic algorithm are:

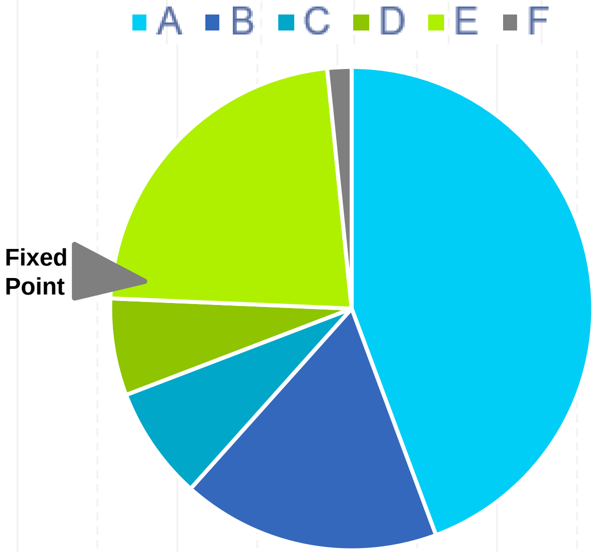
- **Selection:** replicates the most successful solutions found in a population.
- **Recombination (Crossover):** decomposes two distinct solutions and then randomly mixes their parts to form new solutions.
- **Mutation:** randomly changes a candidate solution.

SELECTION

Elitism



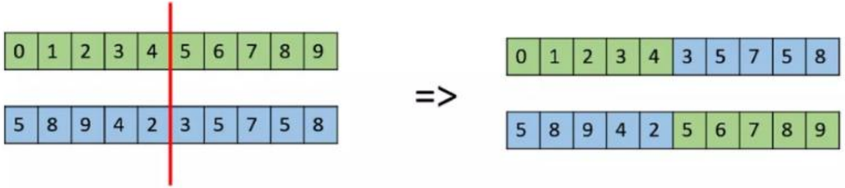
Roulette Wheel Selection



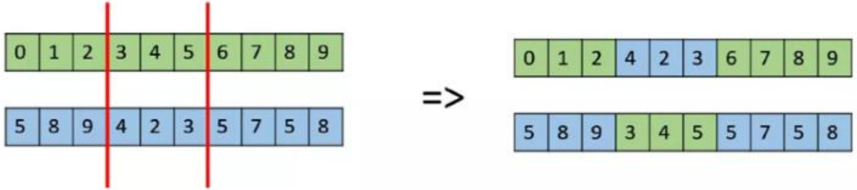
Chromosome	Fitness value
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

CROSSOVER

One-Point Crossover



Multi-Point Crossover



Uniform Crossover

Average Crossover

Environment 5

Define the Environment

The environment is an 11x11 distribution center. As right now there of 4 types of cells in the environment:

Orders (green squares)

The robot can move to these cells and pick up the order.

Aisles (white squares)

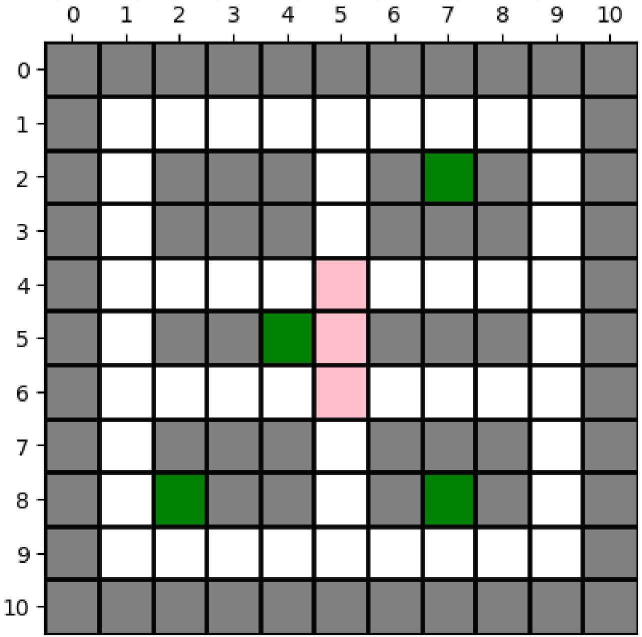
The robot can use them to travel throughout the warehouse

Shelves (gray squares)

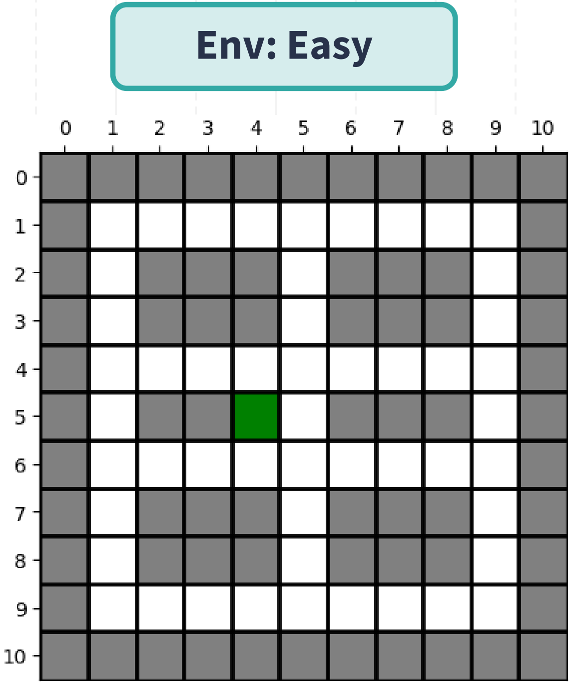
The robot can not move to these cells and these locations are for storing items.

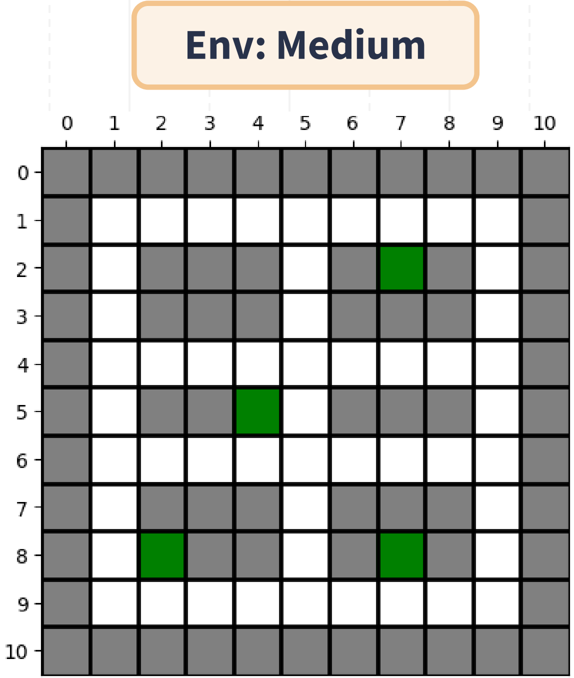
Crowded Area (pink squares)

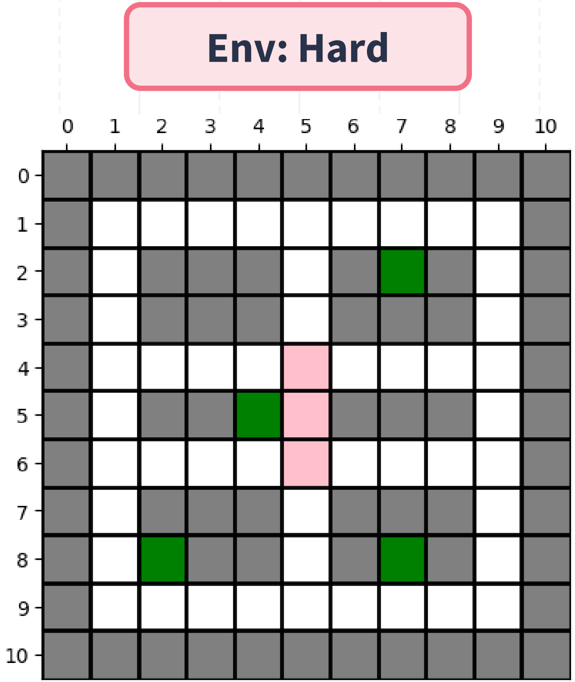
It is better that the robot does not pass through this area (Only in hard environment).



Difficulty Levels







- The robot can move in four directions: **up, down, left, and right.**
- Each order is **only one item**.
- The robot can only move **one step at a time**.
- The agent can only pick up the order if it is in the **same cell** as the order.
- After picking the item from the shelves, the robot must bring the item to a specific location (**starting point**) within the warehouse where the items can be packaged for shipping.

6 Methodology

MAQL-GA

APPROACHES

1. Single-Agent Q-Learning (SAQL)
2. Multi-Agent Q-Learning (MAQL)
3. Multi-Agent Q-Learning – Genetic Algorithm (MAQL-GA)

Rewards

Orders (green squares): +100

We reward the agent with a positive reward if it picks up the order and we finish the episode.

Aisles (white squares) : -1

We punish the agent with a meager negative reward when it passes through these cells.

Shelves (gray squares): -100

We punish the agent with a negative reward if it tries to move to these cells, and then we finish the episode. **(Terminal state)**

Crowded Area (pink squares): -50

We punish the agent with a negative reward so that it does not pass through this area as much as possible.

	0	1	2	3	4	5	6	7	8	9	10
0	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
1	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
2	-100	-1	-100	-100	-100	-1	-100	100	-100	-1	-100
3	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
4	-100	-1	-1	-1	-1	-50	-1	-1	-1	-1	-100
5	-100	-1	-100	-100	100	-50	-100	-100	-100	-1	-100
6	-100	-1	-1	-1	-1	-50	-1	-1	-1	-1	-100
7	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
8	-100	-1	100	-100	-100	-1	-100	100	-100	-1	-100
9	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
10	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100

Rewards

Env: Easy

	0	1	2	3	4	5	6	7	8	9	10
0	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
1	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
2	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
3	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
4	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
5	-100	-1	-100	-100	100	-1	-100	-100	-100	-1	-100
6	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
7	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
8	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
9	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
10	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100

Env: Medium

	0	1	2	3	4	5	6	7	8	9	10
0	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
1	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
2	-100	-1	-100	-100	-100	-1	-100	100	-100	-1	-100
3	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
4	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
5	-100	-1	-100	-100	100	-1	-100	-100	-100	-1	-100
6	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
7	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
8	-100	-1	100	-100	-100	-1	-100	100	-100	-1	-100
9	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
10	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100

Env: Hard

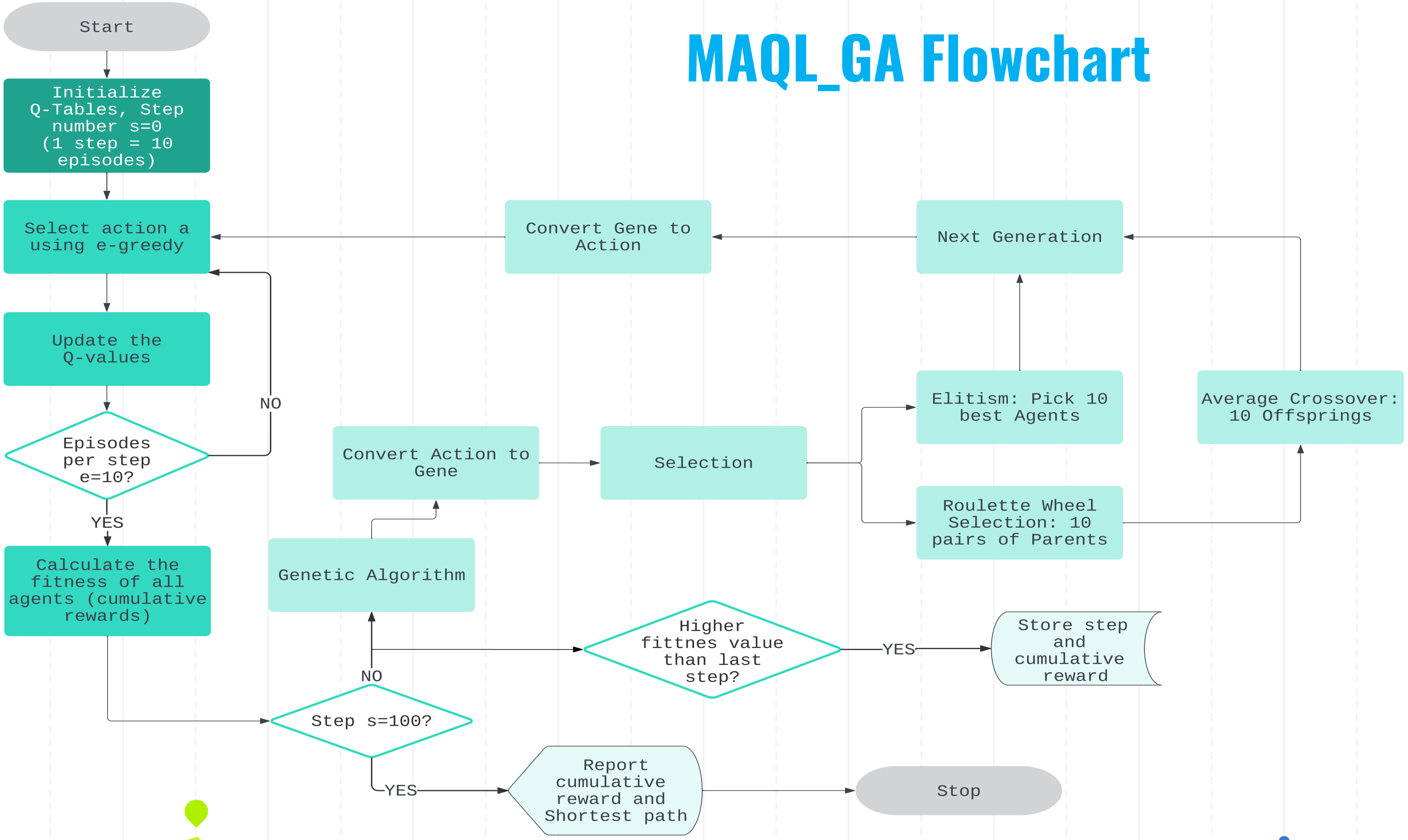
	0	1	2	3	4	5	6	7	8	9	10
0	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100
1	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
2	-100	-1	-100	-100	-100	-1	-100	100	-100	-1	-100
3	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
4	-100	-1	-1	-1	-1	-50	-1	-1	-1	-1	-100
5	-100	-1	-100	-100	100	-50	-100	-100	-100	-1	-100
6	-100	-1	-1	-1	-1	-50	-1	-1	-1	-1	-100
7	-100	-1	-100	-100	-100	-1	-100	-100	-100	-1	-100
8	-100	-1	100	-100	-100	-1	-100	100	-100	-1	-100
9	-100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-100
10	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100	-100

Multi-Agent Approach

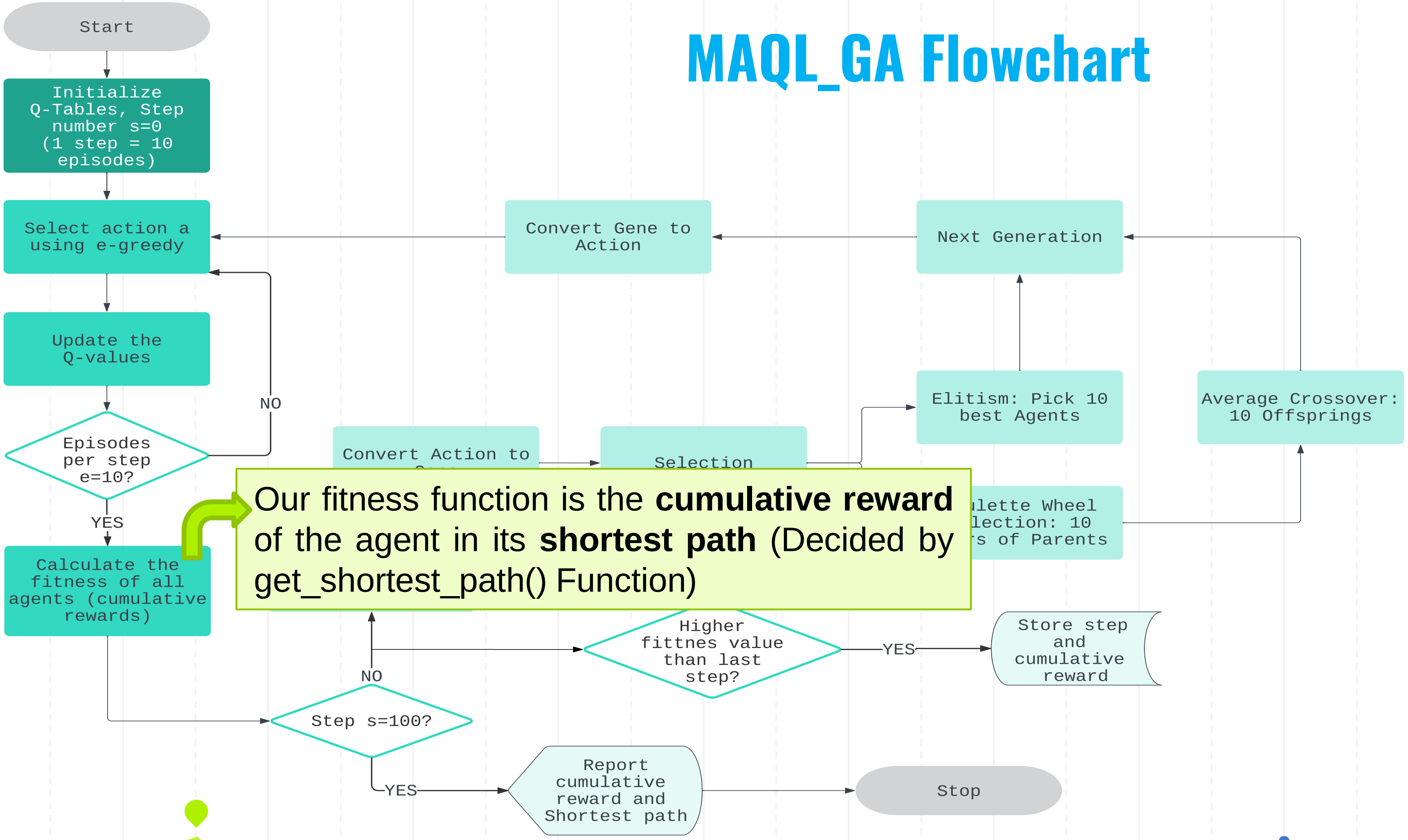
In this approach, we have 20 Q-Agents. Each agent has its own Q-Table, and is able to move in the environment. Agents are trying to find the optimal path to pick up the orders and return to the starting point. We have two approaches to find the optimal path: Centralized and Decentralized.

- Decentralized :Each agent has its own Q-Table and is trying to find the optimal path. These agents don't share their Q-Tables with each other. The best-path is the best-path of the best agent, who has the highest cumulative reward.
- Centralized :Each agent has its own Q-Table and is trying to find the optimal path. These agents share their Q-Tables with each other, through a Genetic Algorithm. After a certain number of episodes, `E`, we combine the Q-Tables of the agents and use the `Genetic Algorithm`.

MAQL_GA Flowchart

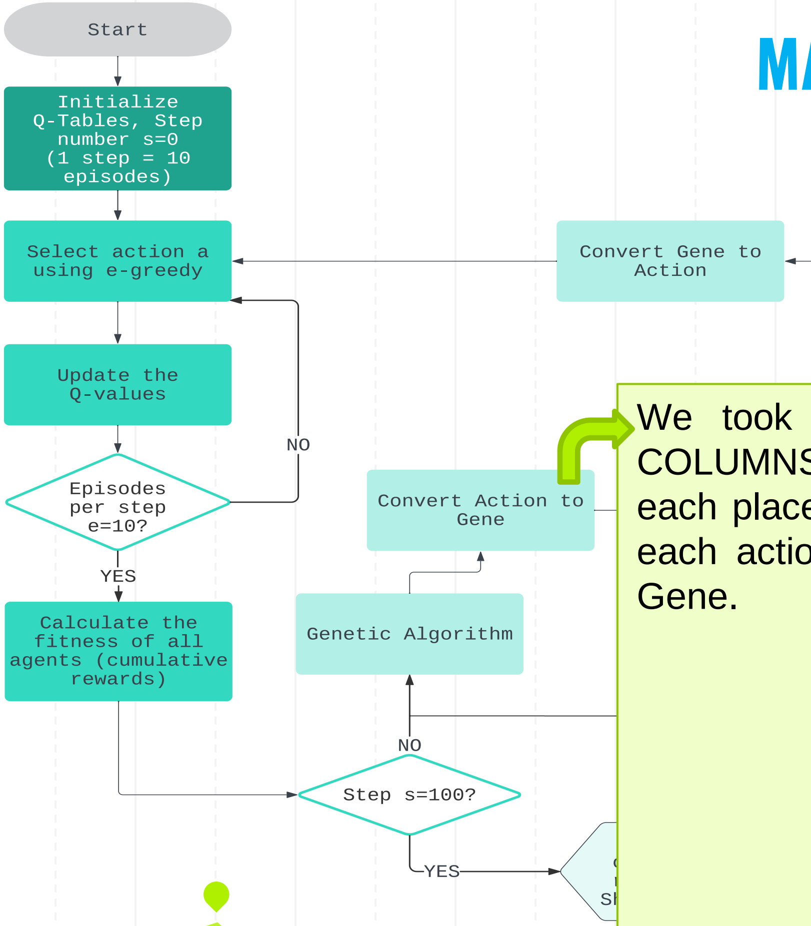


MAQL_GA Flowchart

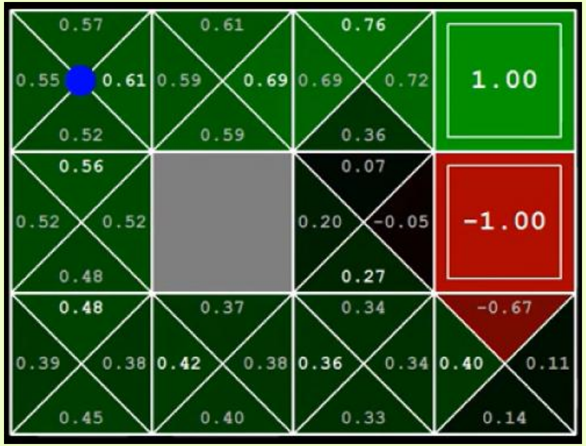


Our fitness function is the **cumulative reward** of the agent in its **shortest path** (Decided by get_shortest_path() Function)

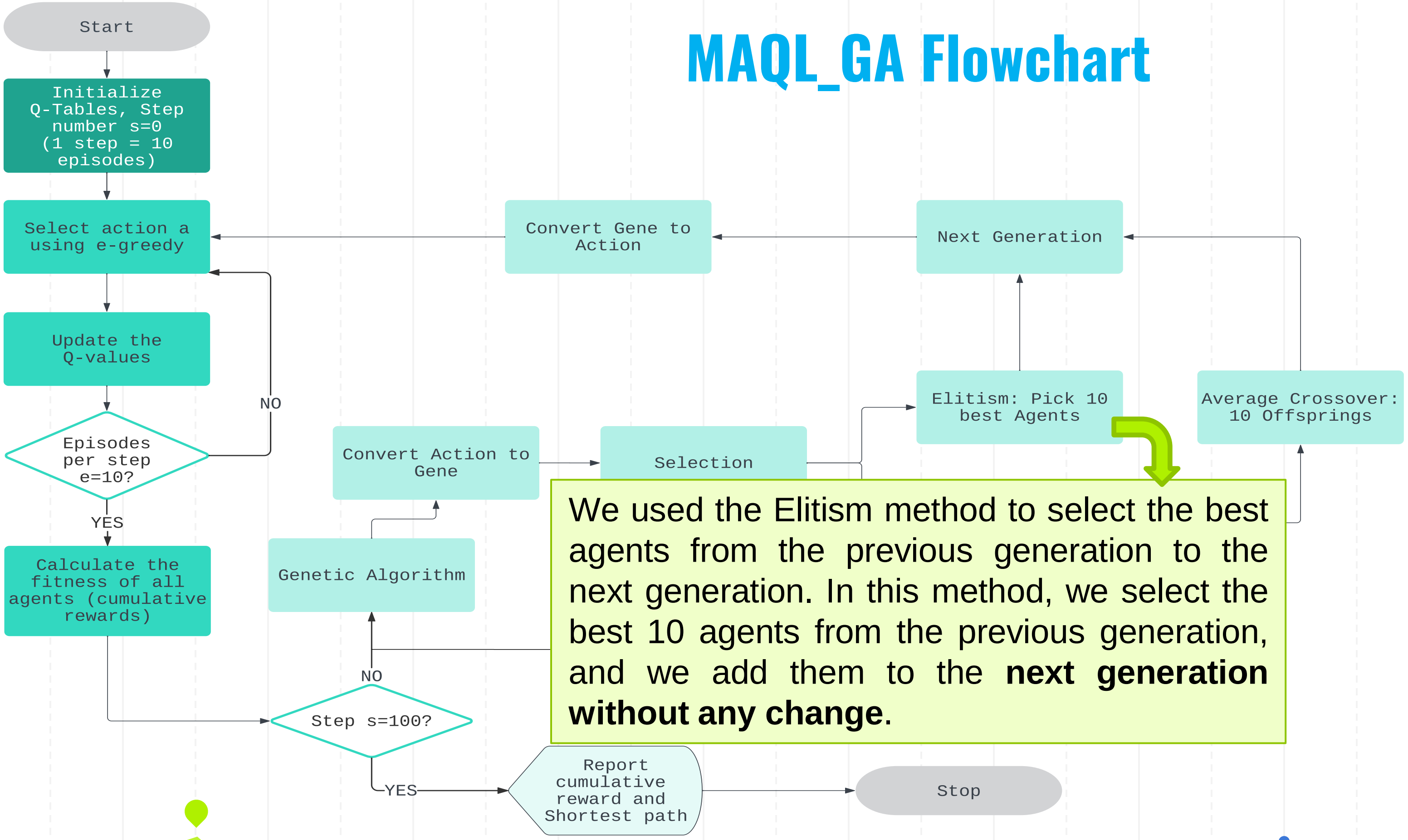
MAQL_GA Flowchart



We took each agents Q-Table with size (ROWS, COLUMNS, ACTIONS) as its Chromosome, where at each place of (ROW, COLUMN) there are 4 values for each action(UP, DOWN, LEFT, RIGHT), which is our Gene.

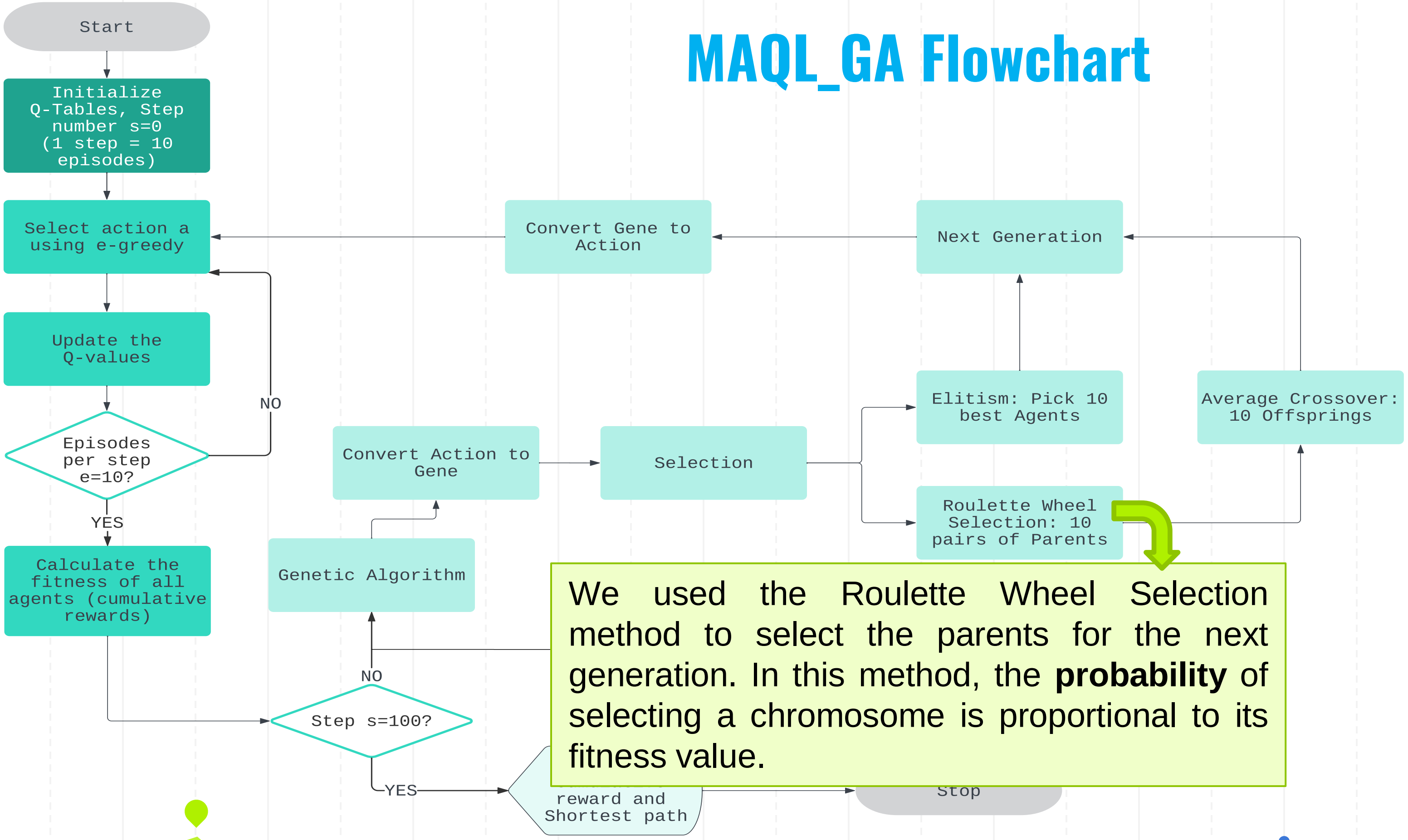


MAQL_GA Flowchart



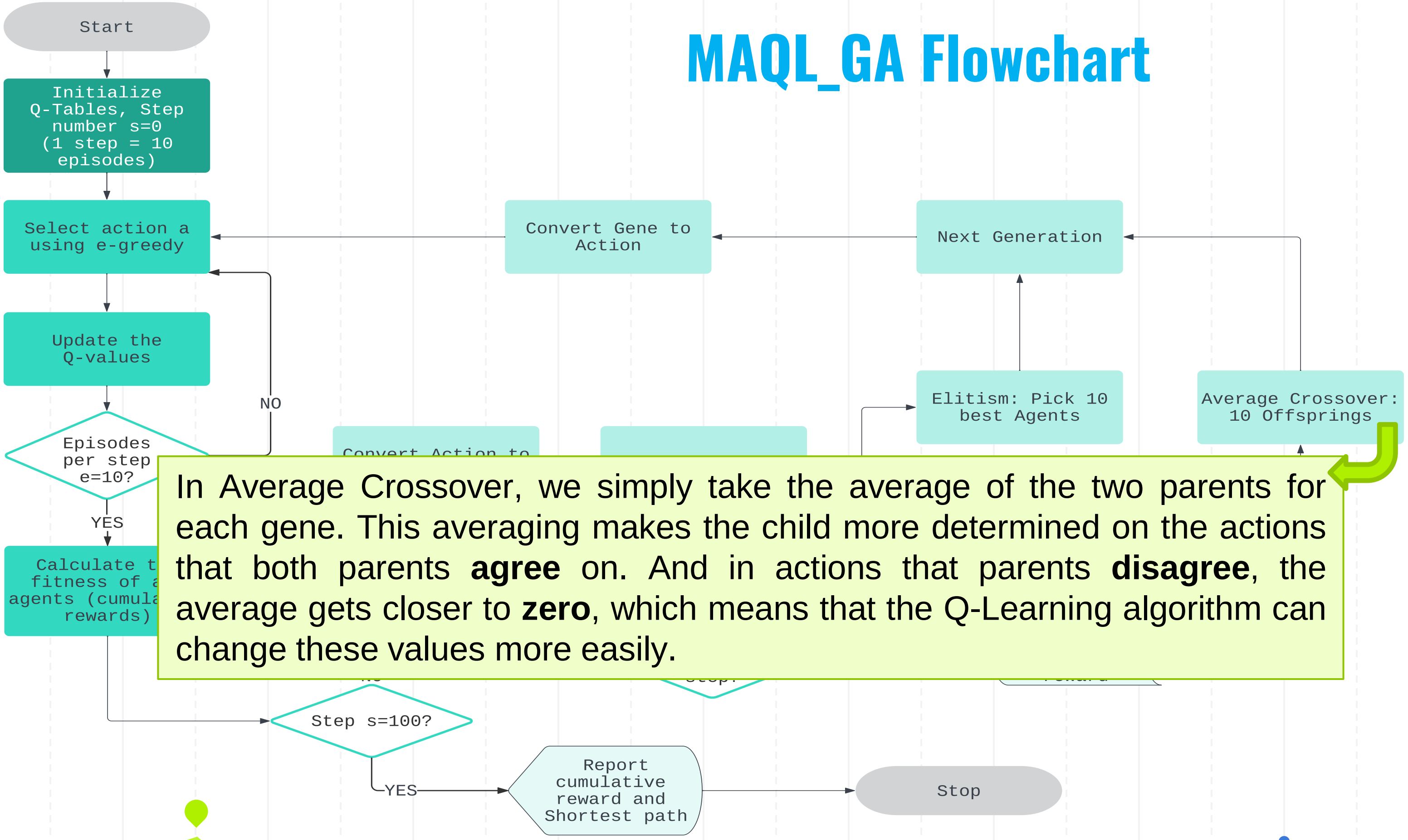
We used the Elitism method to select the best agents from the previous generation to the next generation. In this method, we select the best 10 agents from the previous generation, and we add them to the **next generation without any change**.

MAQL_GA Flowchart



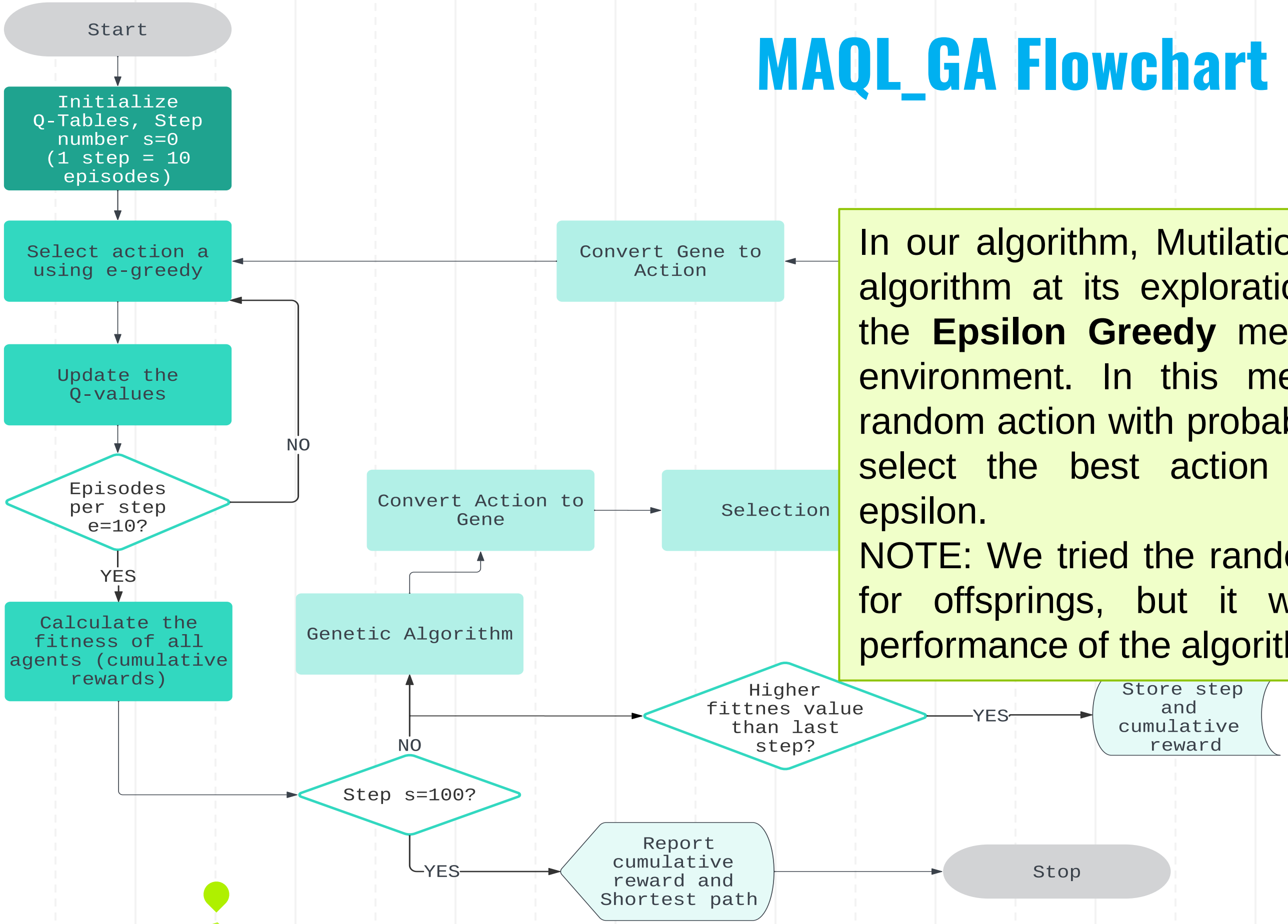
We used the Roulette Wheel Selection method to select the parents for the next generation. In this method, the **probability** of selecting a chromosome is proportional to its fitness value.

MAQL_GA Flowchart



In Average Crossover, we simply take the average of the two parents for each gene. This averaging makes the child more determined on the actions that both parents **agree** on. And in actions that parents **disagree**, the average gets closer to **zero**, which means that the Q-Learning algorithm can change these values more easily.

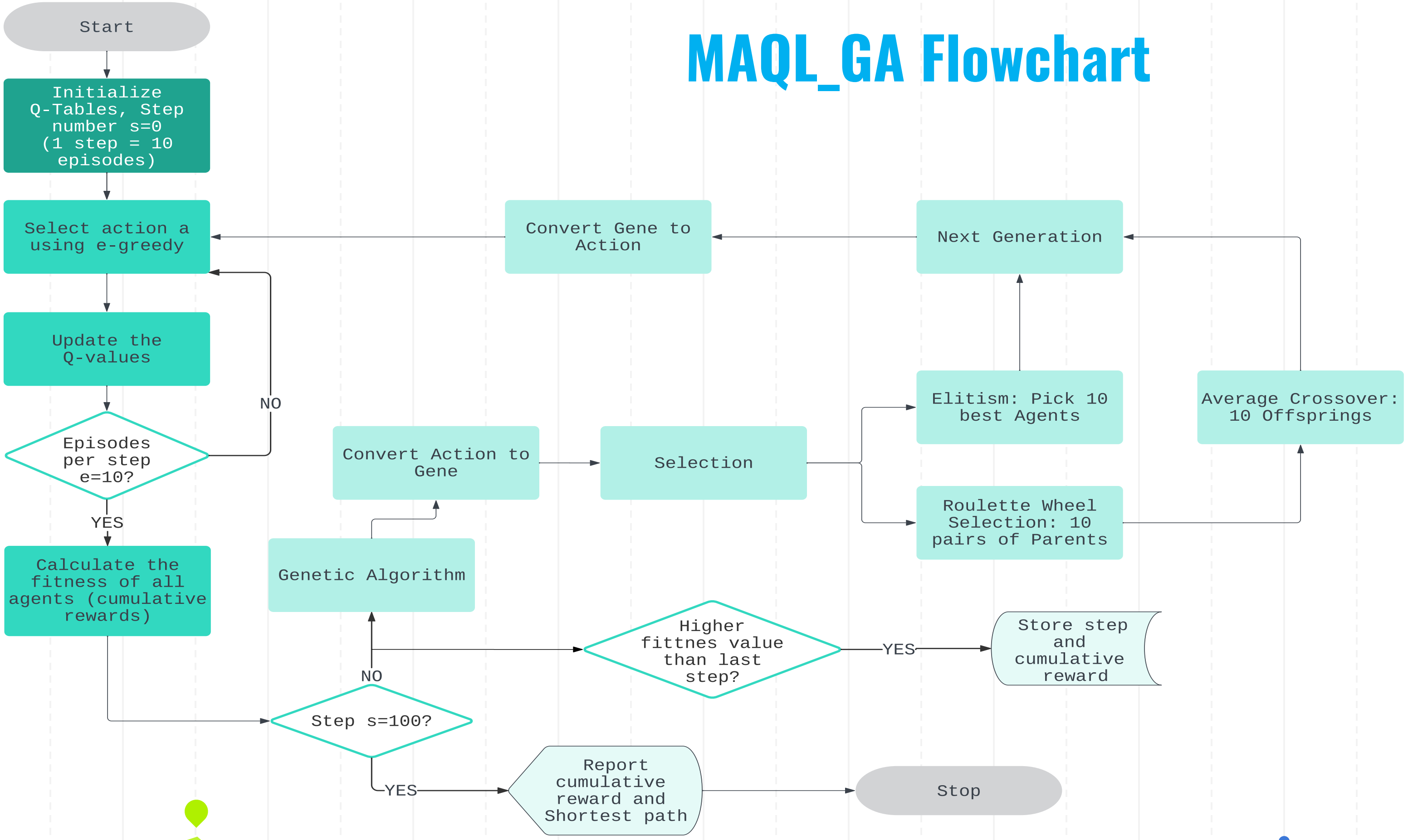
MAQL_GA Flowchart



In our algorithm, Mutilation is our Q-Learning algorithm at its exploration phase. We used the **Epsilon Greedy** method to explore the environment. In this method, we select a random action with probability epsilon, and we select the best action with probability 1-epsilon.

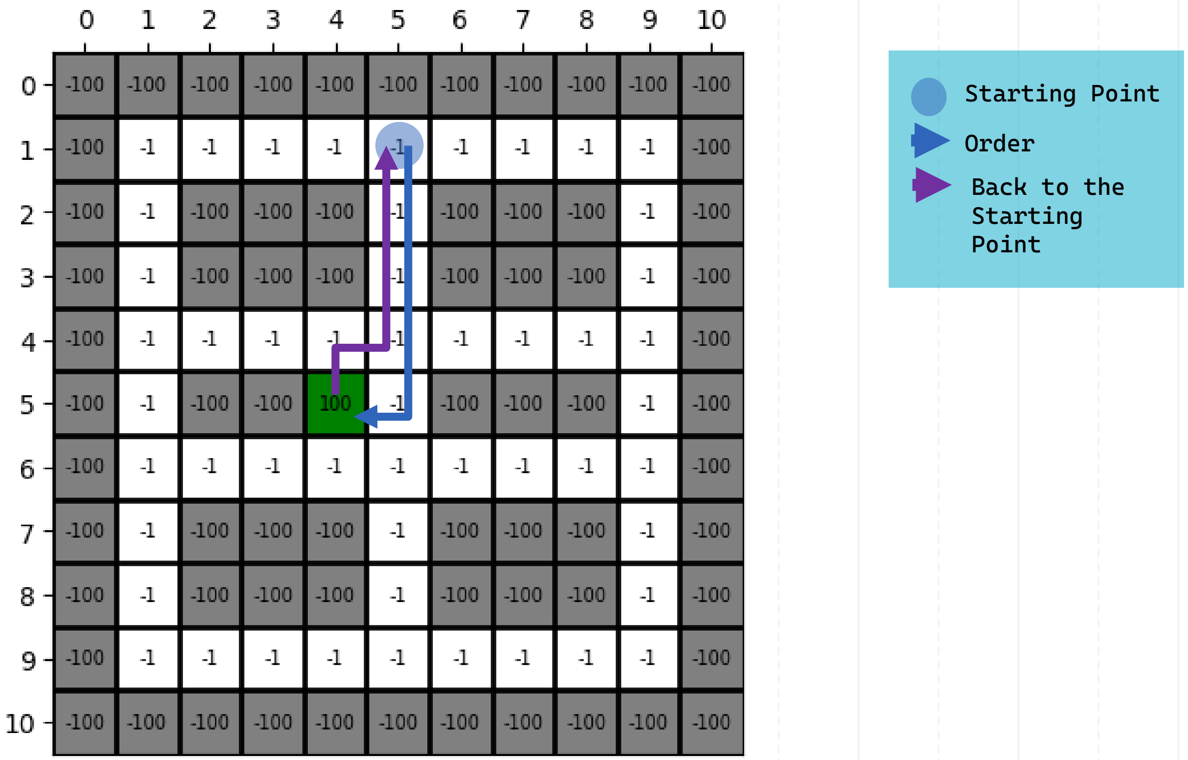
NOTE: We tried the random mutation method for offsprings, but it was decreasing the performance of the algorithm.

MAQL_GA Flowchart

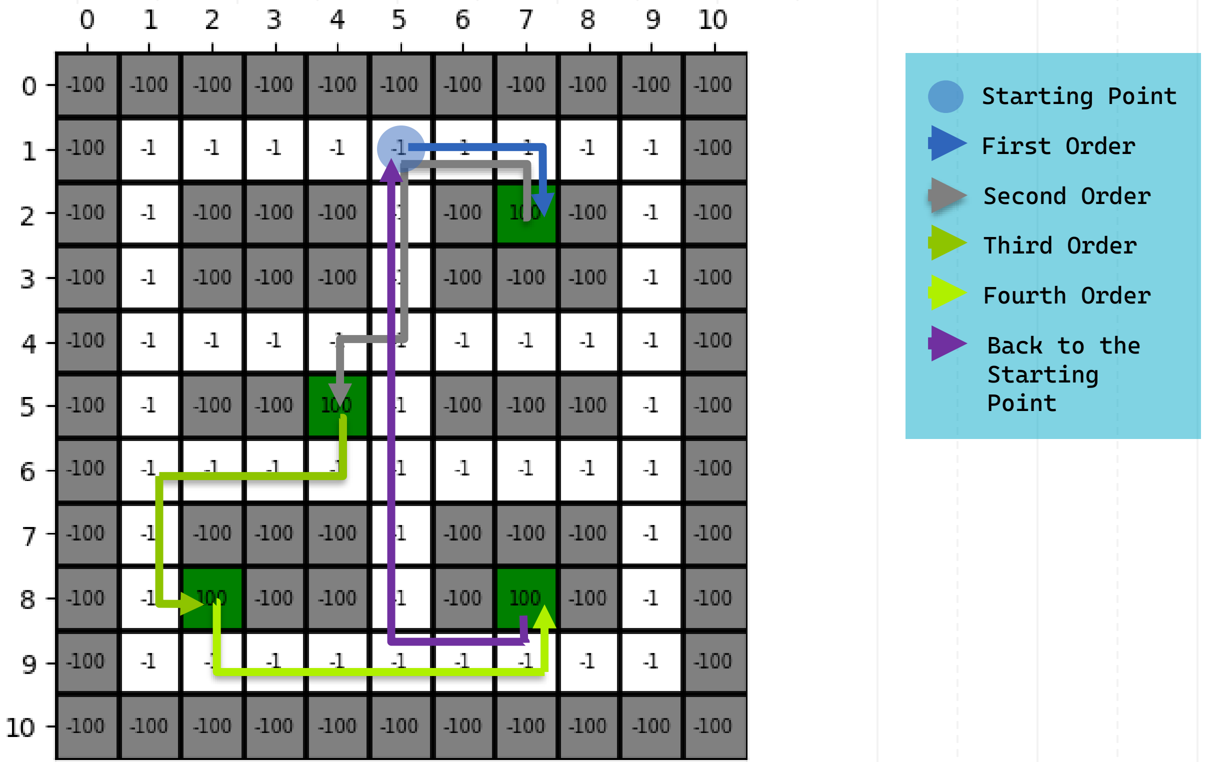


Results 7

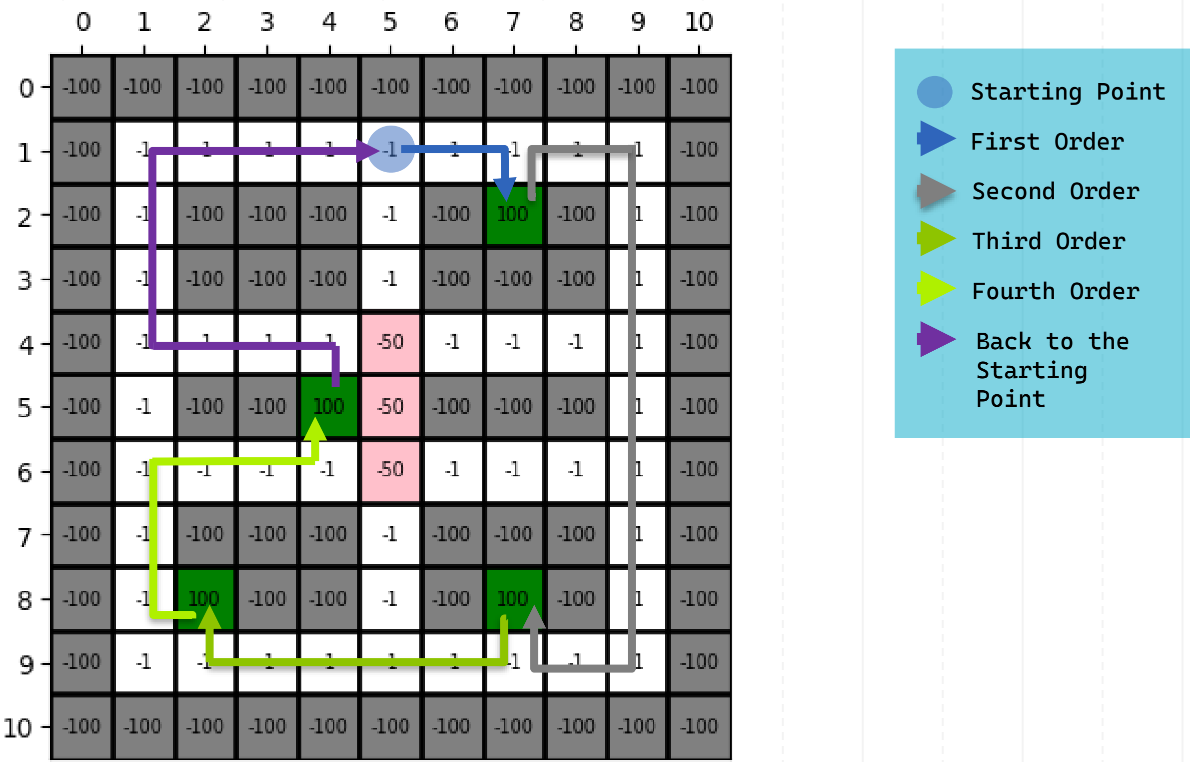
Shortest path – Env: Easy



Shortest path – Env: Medium



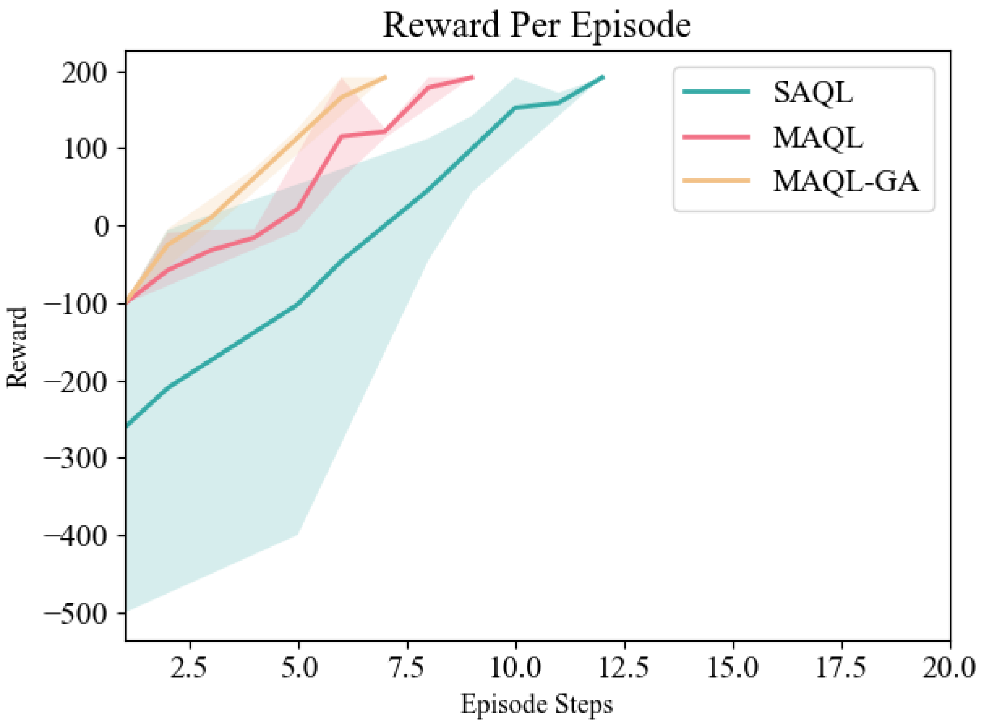
Shortest path – Env: Hard



Env: Easy

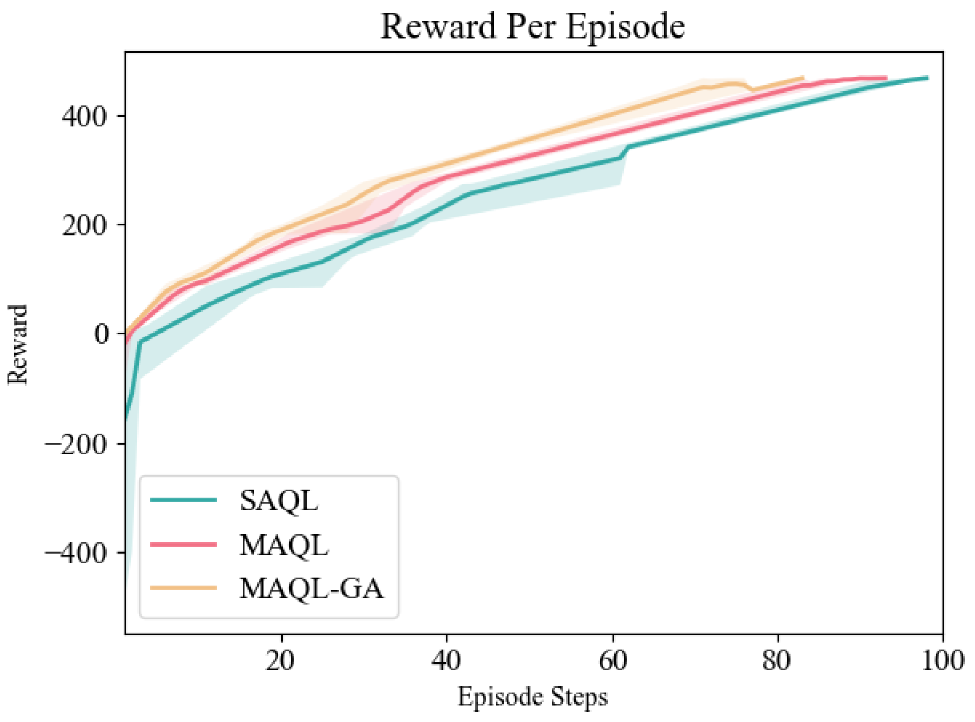
MAQL-GA reached the optimal solution faster, after that, multi-agent Q-learning and finally single-agent Q-learning, which is the slowest.

➢ Why Single Agent Cross Validation plot has a wider range compared to multi agent approaches?



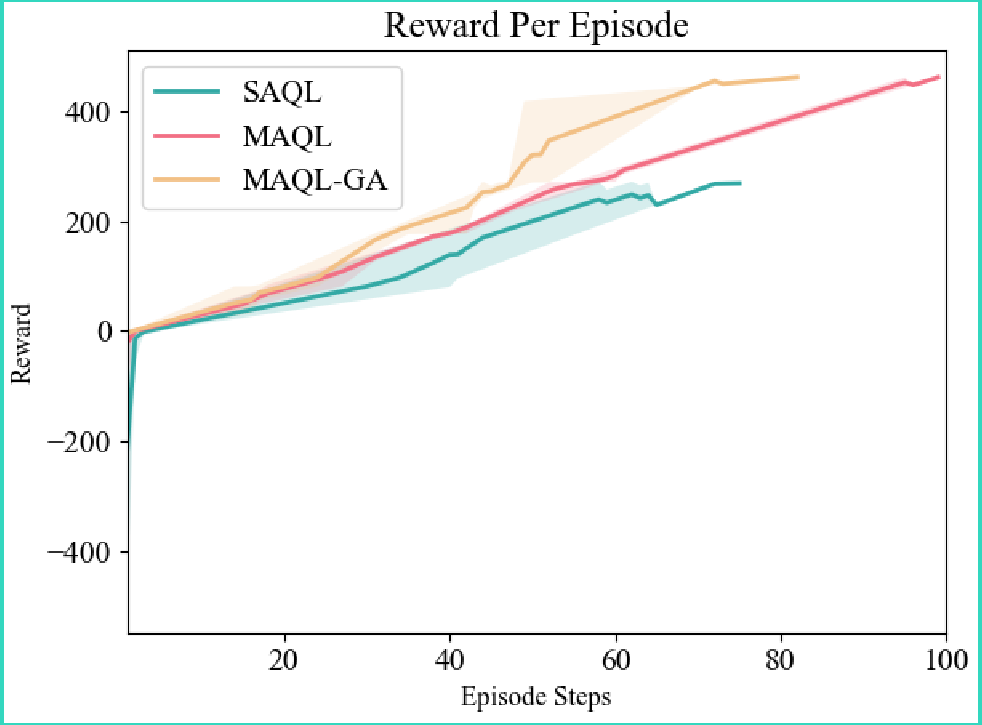
Env: Medium

Like the easy environment, MAQL-GA reached the optimal solution faster, which is in the 80th period, after that, multi-agent Q-learning and finally single-agent Q-learning, which is the slowest.

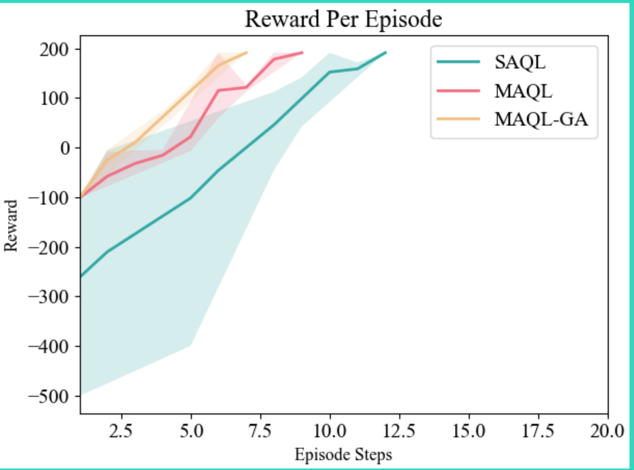


Env: Hard

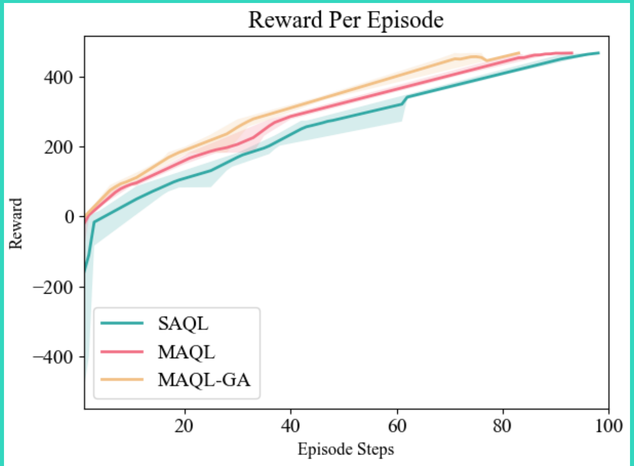
MAQL-GA has reached the optimal solution faster, after that Multi-Agent Q-Learning but Single-Agent Q-Learning was not able to find the optimal solution or even Complete all orders.



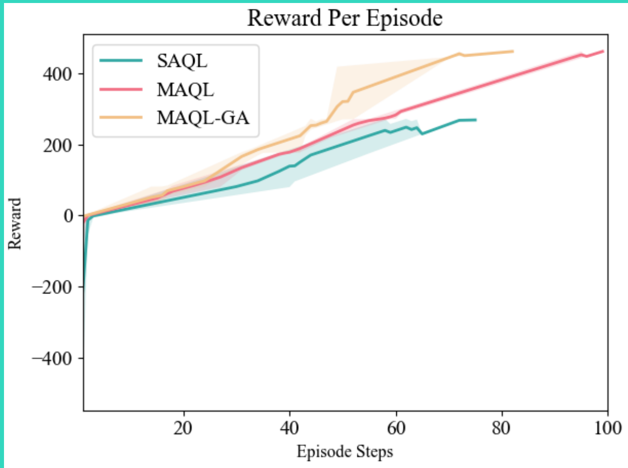
Env: Easy



Env: Medium



Env: Hard



Conclusion 8

Conclusion

This thesis has investigated the integration of multi-agent reinforcement learning (MARL) and genetic algorithms (GAs) to optimize the order picking path in distribution centers. The results have shown the superiority of this approach in the **speed of reaching the optimal path** compared to traditional methods such as Q-learning and multi-agent Q-learning. Due to its higher speed, this algorithm can **reduce the costs and planning time** of collecting orders.

Conclusion

Future research can improve this study by considering **dynamic environments** and **several order picking robots** that collect orders at the same time in the same environment, **dividing their work** and also the **possibility of collision** with each other, as well as **comparing with other centralization methods**. **Expand learning agents**. Additionally, exploring the **scalability** of the approach in larger distribution centers with more complex layouts could provide valuable insight.

THANKS!