Multi-agent Autonomous Drone System

Introdução aos Sistemas Inteligentes e Autónomos

João Silva, up202107600
Leonardo Regadas, up202108144
Diogo Miranda, up202106743

Contexto do problema

O objetivo é implementar um sistema multi-agente que controla drones automáticos que cooperam entre eles para realizar entregas da forma mais eficiente, controlando fatores como o tempo de entrega e ajustes em tempo real.

Para isso, é necessário criar agentes para drones que são responsáveis por realizar uma escolha automática de packages, agentes para um centro de entregas responsável por gerir a entrada e saída de drones e atribui-los packages e estabelecer uma ligação entre eles de forma coordenada para tornar o processo o mais eficiente possível.

Tipos de agentes

Para realizar este projeto, criamos um agente para o centro de entregas que é responsável por gerir 9 agentes de drones.

O delivery hub, constituído por informações dos packages e dos drones, tem como objetivo atribuir os pacotes aos drones e controlar a própria entrada e saída dos mesmos

Os drones, por sua vez, obedecem às ordens do delivery hub e contém atributos como capacidade máxima de peso, velocidade, distância máxima e energia

Interações e protocolos de comunicação

Inicialmente os drones e o delivery hub são identificados no ambiente (área) do problema. Logo de imediato, os drones são atribuídos ao hub em questão, juntamente com as encomendas.

Após o hub atribuir ao drone uma encomenda, o drone informa ao hub quando abandona o local, quando chega ao destino e faz a entrega e quando volta para o hub. Esta última mensagem é responsável por informar ao hub que já concluiu o processo todo e que está disponível para fazer outra encomenda, e assim o hub comunica com outro drone para realizar a próxima entrega.

Estratégias e arquiteturas usadas

Para realizar a distribuição dos packages aos drones, primeiramente verificamos se os drones tinham a capacidade para fazer uma certa entrega:

- capacidade do drone > peso da encomenda
- raio de alcance do drone > distancia da encomenda ao hub
- energia suficiente para executar a próxima entrega (definimos que se após ser feita a entrega a energia fosse abaixo de 25, o drone recarregar-se-ia primeiro, como forma de precaução)

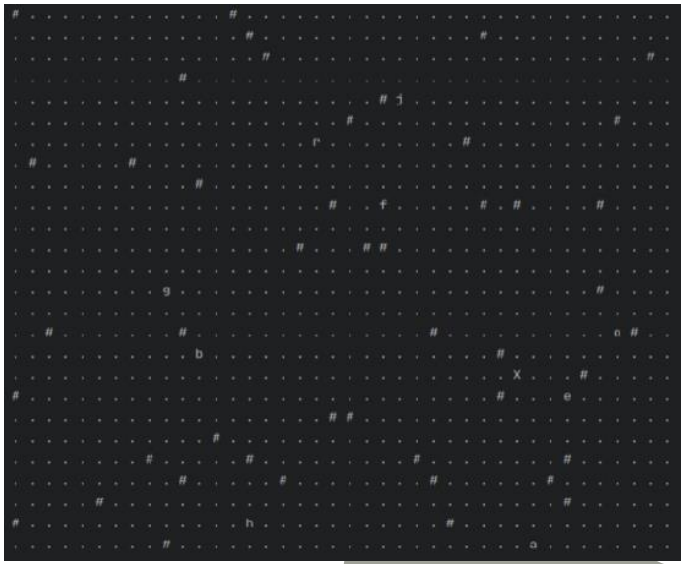
Já no processo de calcular o melhor caminho entre o hub e o destino, recorremos à Euclidean Distance, tendo em conta obstáculos presentes na área que alteram a cada vez que o programa é executado.

Representação dos dados

Para este projeto, utilizamos uma grid de dimensões 40x40, onde cada quadrícula representa um metro.

“.” – lugar vazio
“#” – lugar com obstáculo
“H” – delivery hub
“a” – destino da encomenda “a”

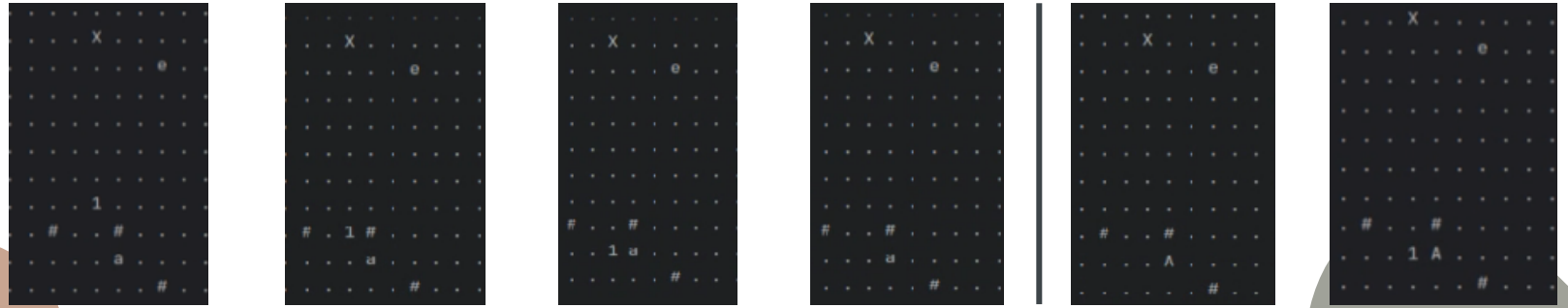
Além disso, quando o drone executa o seu caminho, a grid é atualizada com o número correspondente ao drone e à posição que o mesmo se encontra



Output da grid

É gerada uma nova grid a cada vez que o drone avança uma quadrícula.
Como cada drone tem uma velocidade diferente, diferentes drones geram novas grids a velocidades diferentes, velocidade esta que é calculada dividindo 1 pela velocidade do drone. (Exemplo, se a velocidade do drone é 4 m/s, o grid é atualizado a cada 0.25 segundos)

Entrega



Resultados

Para observar os resultados, decidimos dar print à energia gasta pelo drone, à respetiva energia restante do drone e ao tempo de entrega do package (após o drone regressar ao hub).

Vamos analisar o comportamento do drone 8, por exemplo.

```
drone8 = DroneAgent( jid: "Drone8@localhost", password: "password", hubH, drone_name: "8", max_weight: 40, speed: 4.1, energy_capacity: 140,
await drone8.start()
drones.append(drone8)
```

```
Sent message: Drone drone8@localhost picked package h and left the hub
Sent message: Drone drone8@localhost is delivering the package h to the position (14, 37)
```

```
Energy wasted: 58
Drone drone8@localhost energy left:  82
Delivery time: 8.293 seconds
```

```
Sent message: Drone drone8@localhost recharging...
Sent message: Drone drone8@localhost picked package q and left the hub
Sent message: Drone drone8@localhost is delivering the package q to the position (5, 11)
```

Conclusões

Com este projeto pudemos ter uma melhor noção de como construir um sistema com vários agentes a comunicarem entre si, assim como, num modo geral, a colocar em prática conhecimentos adquiridos para resolver um problema que nos foi proposto.

Não obtemos os resultados que esperávamos, uma vez que existem vários aspetos a melhorar na parte de atribuição de packages aos drones, por exemplo.

No futuro, pretendemos abordar de uma melhor maneira este tipo de problemas, sendo este projeto essencial para um melhor conhecimento da área nunca antes aprofundada o suficiente.

Menu

Backup slides

Protocolos de comunicação

```
Sent message: Drone 1 is ready
Sent message: Drone 2 is ready
Sent message: Drone 3 is ready
Sent message: Drone 4 is ready
Sent message: Drone 5 is ready
Sent message: Drone 6 is ready
Sent message: Drone 7 is ready
Sent message: Drone 8 is ready
Sent message: Drone 9 is ready
Hub hubh@localhost is ready.
Hub hubh@localhost is located at (30, 30)
Assigned Drone 1 to hubH
Assigned Drone 2 to hubH
Assigned Drone 3 to hubH
Assigned Drone 4 to hubH
Assigned Drone 5 to hubH
Assigned Drone 6 to hubH
Assigned Drone 7 to hubH
Assigned Drone 8 to hubH
Assigned Drone 9 to hubH
['drone1', 'drone2', 'drone4', 'drone7', 'drone8', 'drone3', 'drone5', 'drone6', 'drone9']
Creating the environment.
hubH: Managing drones at (30, 30).
hubH: Managing packages at (30, 30).
Sent message: Drone drone1@localhost picked package a and left the hub
Sent message: Drone drone1@localhost is delivering the package a to the position (31, 38)
```

```
Sent message: Drone drone1@localhost returned to HubH
Energy wasted: 30
Drone drone1@localhost energy left:  120
Delivery time: 5.714 seconds
```

```
Sent message: Drone drone1@localhost stopped!
Sent message: Drone drone2@localhost stopped!
Sent message: Drone drone3@localhost stopped!
Sent message: Drone drone4@localhost stopped!
Sent message: Drone drone5@localhost stopped!
Sent message: Drone drone6@localhost stopped!
Sent message: Drone drone7@localhost stopped!
Sent message: Drone drone8@localhost stopped!
Sent message: Drone drone9@localhost stopped!
[]

Process finished with exit code 0
```

Teste - exemplo

Drone	Max_weight	Speed	Energy	Range
1	40	3.5	150	60
2	40	2.2	140	40
3	20	4.7	180	30
4	40	3.1	150	80
5	20	3.9	190	50
6	20	2.4	200	45

Teste - exemplo

Considerando uma grid 40x40 com hub em (30, 30) e packages:

Nome	a	b	c	d	e	f	g	h	i
Pos	(31, 38)	(11, 29)	(17, 4)	(24, 6)	(33, 31)	(22, 22)	(9, 26)	(14, 37)	(15, 1)

Resultados

```
Sent message: Drone drone1@localhost returned to HubH
Energy wasted: 30
Drone drone1@localhost energy left:  120
Delivery time: 5.714 seconds
```

```
Sent message: Drone drone2@localhost returned to HubH
Energy wasted: 56
Drone drone2@localhost energy left:  84
Delivery time: 15.0 seconds
```

```
Sent message: Drone drone3@localhost returned to HubH
Energy wasted: 90
Drone drone3@localhost energy left:  90
Delivery time: 10.638 seconds
```

```
Sent message: Drone drone4@localhost returned to HubH
Energy wasted: 72
Drone drone4@localhost energy left:  78
Delivery time: 13.226 seconds
```

```
Sent message: Drone drone5@localhost returned to HubH
Energy wasted: 20
Drone drone5@localhost energy left:  170
Delivery time: 3.846 seconds
```

```
Sent message: Drone drone6@localhost returned to HubH
Energy wasted: 44
Drone drone6@localhost energy left:  156
Delivery time: 11.25 seconds
```