Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №2

“Ознайомлення з середовищем Jupyter Notebook”

З дисципліни

“Вступ до Штучного Інтелекту”

Виконав:
Студент групи ІП-05
Амелін В. О.

Перевірив:
Таран В. І.

Київ - 2022

Мета роботи: розробити інтелектуального агента-машину, що рухається по графу-дорозі з попередньої лабораторної роботи за заданими правилами та метою. Отримати практичні навички роботи з інтелектуальними агентами.

Завдання:

- Отримати навички роботи з інтелектуальними агентами.
- Розробити раціонального агента-автомобіль, що рухається по дорозі з попередньої лабораторної роботи.

Опис алгоритму генерації дороги

Сам граф спочатку генерується за допомогою методу бібліотеки networkx - grid_2d_graph.

Лабораторна робота заснована на першій роботі, де ми створювали зв'язний граф і видалили в ньому ребра. Алгоритм працює за таким принципом:

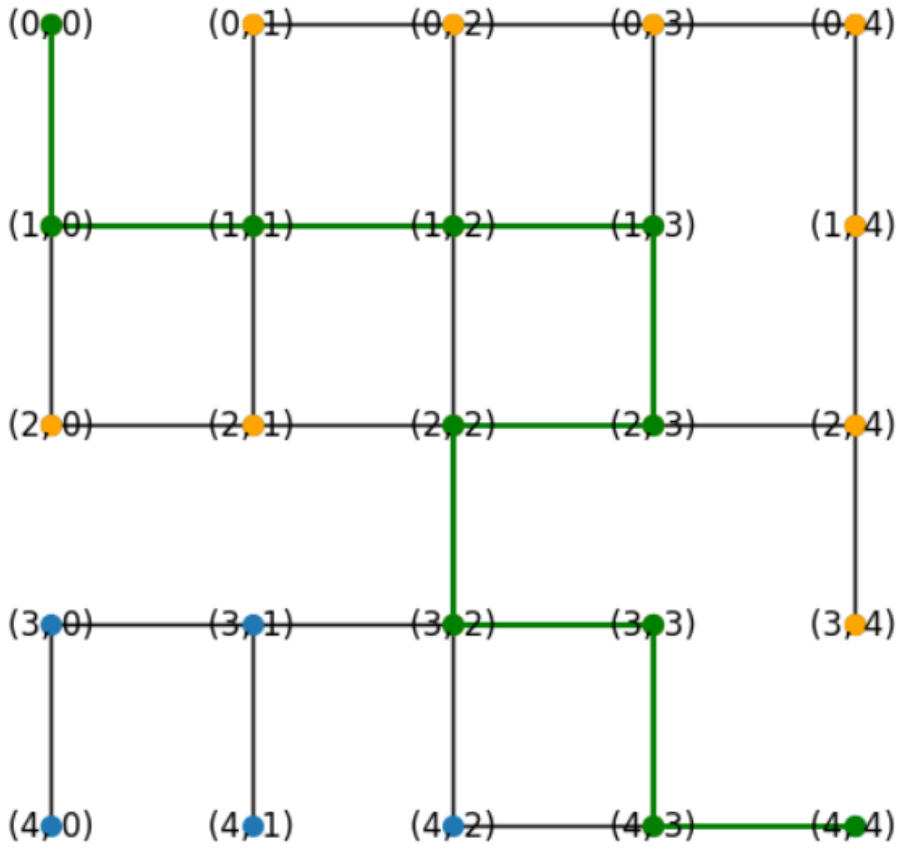
1. Ми перевіряємо коректність введених точок.
2. Додаємо в масив відвіданих точок і масив шляху.
3. Йдемо в наступну вершину
4. Перевіряємо, чи є ця нода і суміжні до них виходом.
5. У позитивному випадку виходимо з циклу.
6. У негативному випадку, якщо ми зайшли в глухий кут, ми видаляємо останнє значення в масиві шляху.

7. **Програмний код генерації дороги:**

```
start_cord = (0, 0)
end_cord = (4, 4)
if((size, size) <= end_cord and (size, size) <= start_cord):
  raise ValueError(f"Should be less {size}")

if(end_cord > (0, 0) and start_cord > (0, 0)):
  raise ValueError("Must be greater than zero")
def agent(G, start_cord, end_cord):
  visited = (start_cord, )
  path = [start_cord]

  while path[-1] != end_cord:
    next_nodes = G.adj[path[-1]]
    best = None

    for node in list(next_nodes):
      if node == end_cord:
        path.append(node)
        print(*path, sep = ' => ')
        return path, visited
      elif node not in visited:
        if best is None or abs(node[0] - end_cord[0]) + abs(node[1] -
end_cord[1]):
          best = node

    if best is None:
      path.pop()
    else:
      path.append(best)
      visited += (best,)

    print(*path, sep = ' => ')

path, visited = agent(G, start_cord, end_cord)
plt.figure(figsize=(5,5))
pos = {(x,y):(y,-x) for x,y in G.nodes()}
nx.draw(G, pos, with_labels=(pos, pos), node_size=50)
```

```
nx.draw(G, pos, nodelist=visited, node_color='orange', node_size=50,)
nx.draw(G, pos, nodelist=path, node_color='g', node_size=50,
edgelist=[(path[i], path[i+1]) for i in range(len(path)-1)],
edge_color='g', width=2)

plt.show()
```

Результат:



Висновок: під час виконання лабораторної роботи, я закріпив отримані знання роботи з юпітер ноутбук. Створив алгоритм знаходження виходу з лабіринту. Отримав навички роботи з агентом.