**#1294388 - #****1311319**

**Using ant colony optimization to solve a Rubik's cube**

**Agents:**

* ants

**Interaction rules:**

* walk randomly
* Follow the path that has more pheroneme ( on the way going to the food)
* Choose the path in a probalistic way based on the level of pheroneme on the way back to the nest)

**Validation:**

Since we have two distinct parts of the project ( 1- Ant colony optimization & 2- Ant colony optimization with cube), we can implement and validate them separately:

**1- Ant colony optimisation**

* Does it converge to an optimal solution ?

Since the cube has a very large search space (~ 4.2 \* 10^19), we can add an upper bound to test it initially. By doing this we decrease the size of the graph greatly – for testing – and we can iteratively increase the upper bound, to test for larger search spaces of the cube:

The size of the graph for an upper bound, n is given by :A description...

Hence, our search space will grow as follows:

|  |  |
| --- | --- |
| **Upper bound** | **Size of the graph (search space)** |
| 1 | 19 |
| 2 | 289 |
| 3 | 4339 |
| 4 | 65089 |
| 5 | 976339 |
| 6 | 14645089 |

And so on ..

We test with *cube*

*states* that have short solutions first, and then try larger ones.

**2- cube**

* Testing with a real cube
* Comparing with a CPU solver

**Hypothesis**

The “ants” (ant colony optimization) will converge to an optimal solution after a sufficient amount of time with a Rubik's cube graph.