README: AI Lab - Ant Colony Optimization

This document details the development process and findings of the Artificial Intelligence laboratory, focused on route optimization using an Ant Colony Optimization (ACO) algorithm.

Phase 1: Basic Functionality

In the initial phase, the main objective was to establish the basic functionality of the program. This was achieved as follows:

- **Board and Pheromone Implementation:** A functional board was developed that allowed ants to deposit pheromones.
- **Pathfinding:** The logic was implemented so that the ants could find the best possible path across the board.
- **Sequential Execution:** In this first stage, the entire process was executed sequentially, using a single processing thread.

Phase 2: Hyperparameter Optimization

Once the basic functionality was operational, the next step was to optimize the algorithm's hyperparameters to improve the results. This process was carried out in several stages:

1. **Manual Adjustment:** Initially, an attempt was made to adjust the hyperparameters manually. However, this approach did not produce satisfactory results.
2. **Grid Search:** A Grid Search was implemented to systematically explore different combinations of hyperparameters. Despite being a more structured method, the results were not as expected.
3. **Random Search:** Next, a Random Search was used, which offered a slight improvement in results compared to the previous methods.
4. **Genetic Algorithm:** Finally, a genetic algorithm was implemented that used the random search as a basis. This approach proved to be the most effective, starting to deliver significantly better results.

Phase 3: Performance Improvement

With the genetic algorithm delivering good hyperparameters, it was identified that the simulations to evaluate these parameters were too short. To further enhance the results, the following improvements were made:

- **Increased Simulation Time:** Once the genetic algorithm found a good set of hyperparameters, the simulation was allowed to run for a longer period. This allowed for better convergence towards the optimal solution.
- **Implementation of Parallelism:** It was observed that extending the simulation time significantly slowed down the process. To solve this, parallelism was implemented at the ant level. This allowed the ants to

travel the paths much more quickly, achieving a more efficient convergence towards an optimal result.

Phase 4: Results and Lessons Learned

After completing the optimization and performance improvement phases, the following results and conclusions were reached. The average execution time for a complete optimization run, including hyperparameter search and the final simulation, was approximately **10 minutes**.

- **Best Result:** The best result obtained was approximately **2920**.

Convergence and Stagnation

A key observation was the algorithm's convergence behavior. While significant improvements were made in the early stages of the simulation, the solution tended to stagnate after reaching a certain threshold. This indicates that the algorithm quickly finds a near-optimal solution, but further refinements are marginal.

The following table illustrates a typical convergence pattern during a final 10-minute run:

Time (minutes)	Best Path Length Found
1	4500
2	3800
4	3150
6	2980
8	2925
10	2920

Lessons Learned:

1. **Initial Pheromone Level and Tau-Zero:** Through the genetic algorithms, it was discovered that the optimal values for the initial pheromone level and for `tau_zero` (the base level of pheromones on an edge) are **zero**. The reason is that if you start with a certain level of pheromones, the ants assume that a valid path already exists, when in fact it does not. It is most effective for the only pheromones present on the map to be those deposited by ants that have actually traveled a path.
2. **Number of Ants:** It was observed that the algorithm's performance does not improve linearly with an increase in the number of ants. Experiments showed that approximately **200 ants** is an optimal limit. With a larger number of ants, the algorithm tends not to converge on a good result, as the ants get "lost" and end up exploring very long and inefficient paths.