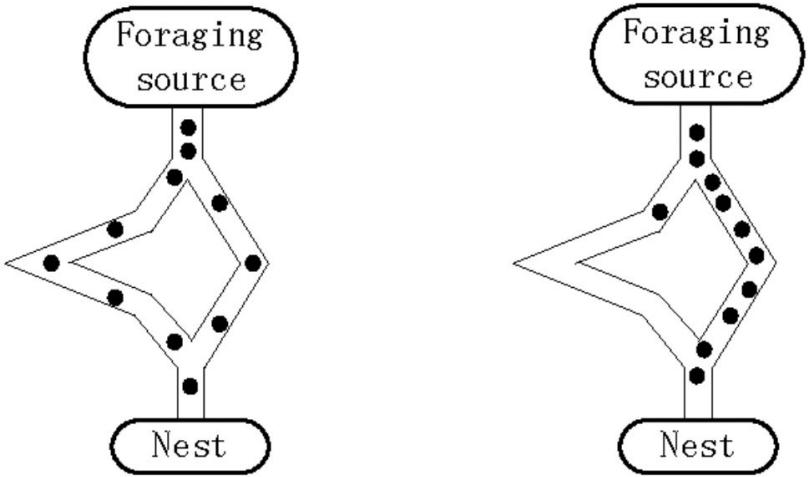
Applications of Ant Colony Optimization

Jacob Epstein

Situation: You have to approximate an optimal solution to an NP-Hard problem.

What would ants do?

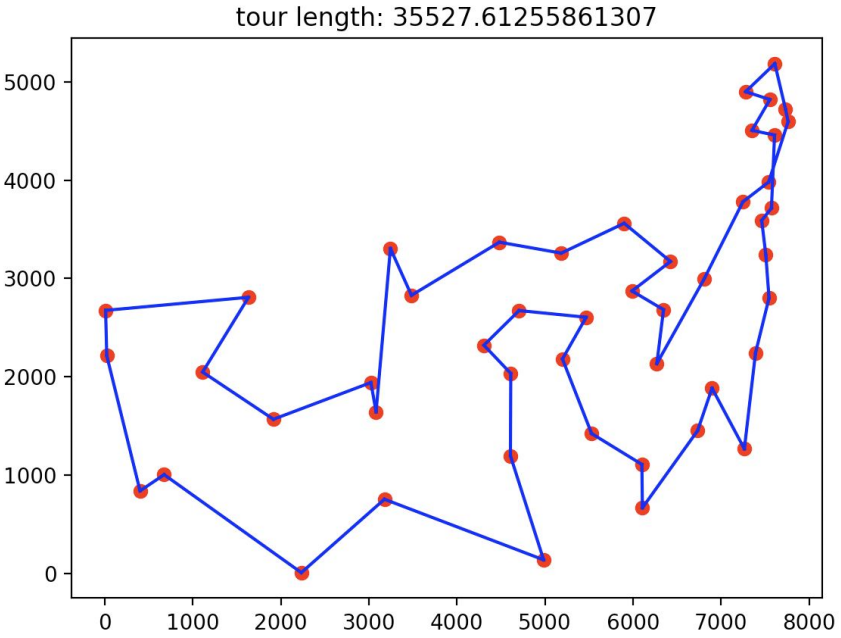
- When searching for a food source, ants deposit **pheromones** on the ground
- Ants are more likely to navigate in a direction that has a higher pheromone concentration
- Leads to the ants finding the most efficient path!

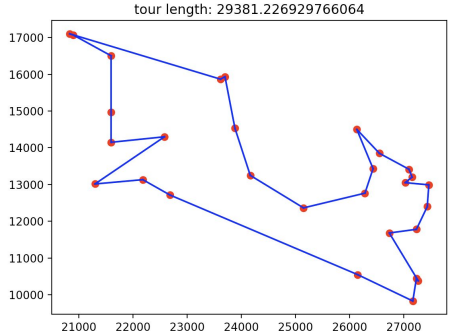


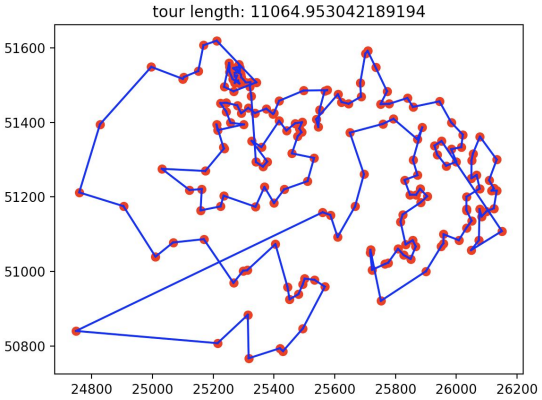
The ACO Algorithm

```
procedure ACO is
    initialize pheromone matrix
    loop
        loop
            each ant applies a state transition rule to incrementally build a solution
            apply local pheromone updating rule
        until all ants have built a complete solution
        apply global pheromone updating rule
    until end_condition
end procedure
```

Application I: Travelling Salesman Problem (Live Demo!)

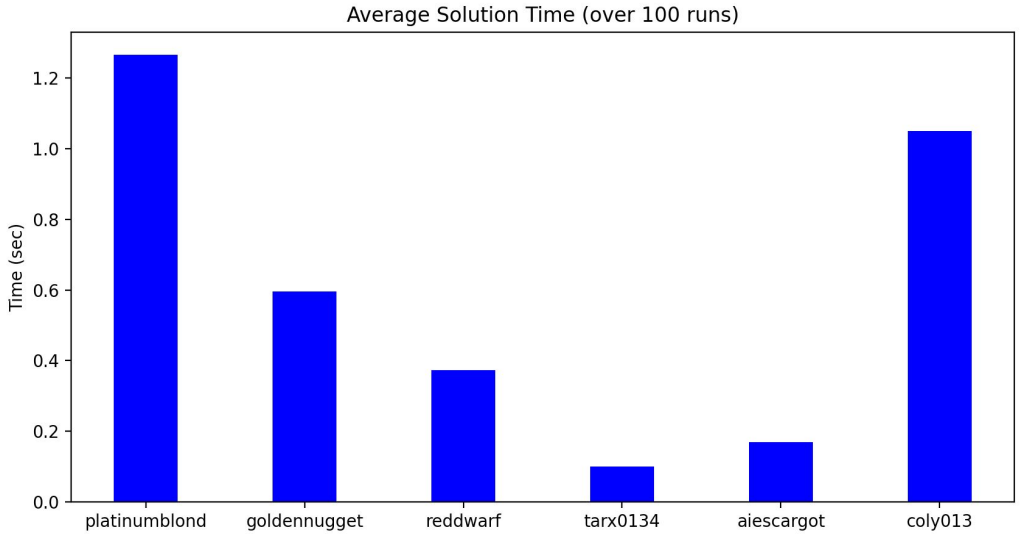






Entries in the pheromone matrix for every edge.

Application II: Sudoku Solver




```
solving aiescargot.txt
1  6  2  8  5  7  4  9  3
5  3  4  1  2  9  6  7  8
7  8  9  6  4  3  5  2  1
4  7  5  3  1  2  9  8  6
9  1  3  5  8  6  7  4  2
6  2  8  7  9  4  1  3  5
3  5  6  4  7  8  2  1  9
2  4  1  9  3  5  8  6  7
8  9  7  2  6  1  3  5  4
solved! took 5 iterations and 0.05035281181335449 seconds.
```

Entries in the pheromone matrix for every possible value of every cell.

Conclusions

- Benefits of ACO
  - An idea that generalizes to a vast amount of problems
  - Quickly generates near-optimal solutions to NP-Hard problems
- Drawbacks of ACO
  - Often gets stuck in local minima
  - No guarantees about the optimality of the solution (or how close to optimal the solution will be)
  - Very sensitive to changes in hyperparameters
- Ways to mitigate these drawbacks
  - Local Search
  - Constraint propagation
  - Pheromone evaporation

Thank you all for a great semester!

Code at https://github.com/jacobe90/aco-applications