Ho Chi Minh City University of Technology (HCMUT)

Advanced Algorithms – Assignment Report

Ant Colony Optimization for Job-Shop Scheduling Problem

Submitted by:
Felix Luca Krebs (2470475)
MD KAMRUZZAMAN RUSSEL (2470478)
Justus Izuchukwu Onuh (2470477)

Submission Date: April 20, 2025

Introduction

Problem Statement

The Job-Shop Scheduling Problem (JSSP) is a widely studied problem in combinatorial optimization. It involves assigning a set of jobs, each consisting of a specific sequence of operations, to a set of machines. Each job must follow a strict operation order, and each operation requires a specific machine for a given duration. The challenge is to find a schedule that minimizes the **makespan**—the total time to complete all jobs.

JSSP is significant in many industries but computationally difficult—it is an **NP-hard** problem. The number of feasible schedules grows exponentially with problem size, making brute-force approaches impractical. Efficient algorithms, especially heuristics and metaheuristics like Ant Colony Optimization (ACO), are required to find near-optimal solutions in a reasonable amount of time.

Applications

JSSP has a wide range of real-world applications:

- **Manufacturing**: Optimizing the workflow of factories where multiple jobs require different machines.
- **Cloud Computing**: Assigning computational jobs to processing nodes to balance loads.
- **Robotics**: Planning robot actions in environments with shared resources.
- **Healthcare**: Scheduling operations and patient diagnostics using shared medical equipment.

Existing Solutions

Traditional techniques like Branch-and-Bound, Dynamic Programming, and Integer Programming work well for small problem instances but are computationally expensive for large problems.

Metaheuristics like Genetic Algorithms, Tabu Search, Simulated Annealing, and Ant Colony Optimization are widely used for large instances. These offer near-optimal solutions in reasonable time frames.

Why Use Ant Colony Optimization (ACO)?

ACO is inspired by the behavior of real ants that use pheromone trails to find the shortest path to food. It is ideal for solving NP-hard problems like JSSP due to:

- **Distributed search**: Multiple agents (ants) explore solutions in parallel.
- **Adaptive learning**: Pheromone trails are reinforced by better solutions.

- **Balance of exploration and exploitation**: It can escape local optima and converge to good solutions.

Methodology

Overview of ACO

ACO models the problem as a graph. Ants construct solutions probabilistically, guided by pheromones (historical learning) and heuristics (domain knowledge).

Applying ACO to JSSP

- **Nodes**: Represent operations within a job.
- **Edges**: Represent valid operation sequences that satisfy precedence and machine constraints.
- **Pheromone Trails (τ_{ij})**: Indicate the desirability of moving from operation i to j.
- **Heuristic Info (η_{ij})**: Often the inverse of processing time.

Transition Probability:

$$P_{ij}^{(k)}(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}(t)]^\beta}{\sum_{l \in N_i^k} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}(t)]^\beta}$$

Pheromone Update:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \Delta\tau_{ij}$$

Where:

- ρ: Evaporation rate
- $\Delta\tau_{ij}$: Sum of contributions from all ants

Experiments

Implementation Details

Language: Python
Repository: `https://github.com/addejans/ACO-JSSP`
Parameters:

- Ants: 20
- Iterations: 100
- α: 1.0

- β: 2.0
- ρ: 0.5

Dataset: FT06 Benchmark

- 6 jobs, 6 machines
- 36 operations in total
- Known optimal makespan: 55

Results and Evaluation

Run	Makespan	Time (s)
1	56	3.2
2	55	2.9
3	55	3.0

Table 1: ACO Results on FT06 Dataset

Average Makespan: 55.3
Conclusion: ACO reliably found optimal or near-optimal schedules within reasonable time.

Improvements and Applications (Bonus)

Possible Improvements

- Incorporate local search (e.g., 2-opt) for refining schedules
- Use dynamic parameters to adapt exploration-exploitation trade-off
- Implement parallel ACO for faster computation

Real-World Extensions

- Smart factories with IoT-enabled scheduling
- Automated hospital scheduling for surgery rooms
- Adaptive cloud job scheduling in high-performance computing

References

1. G. Zhang et al., "Ant Colony Optimization for the Job-Shop Scheduling Problem," arXiv:1309.5110

2. M. Dorigo and T. Stützle, "Ant Colony Optimization," MIT Press, 2004
3. ACO-JSSP GitHub Repo: `https://github.com/addejans/ACO-JSSP`
4. K. M. O. Raza, Py-ACO Library: `https://github.com/KMORaza/Py-ACO`