King Saud University
College of Computer and Information Sciences
Department of Computer Science

Ant Colony Optimization for Graph Coloring Problem

Population-based Metaheuristic

Mohammed Edris Mahdy (446910613)
Mohammed Ahmed Ewida (446910614)

Under the Supervision of:
Prof. Manar Hosny

Outline

Graph Coloring Problem (GCP)

Formal Definition

Given an undirected graph $G = (V, E)$, find a k-coloring function $C : V \to \{1, 2, \ldots, k\}$ such that:

- $\forall (u, v) \in E : C(u) \neq C(v)$ (adjacent vertices have different colors)
- Minimize $k = \chi(G)$ (chromatic number)

Objective Function

$$\text{Minimize } k \quad \text{subject to} \quad F(C) = \sum_{(u,v)\in E} \mathbb{I}(C(u) = C(v)) = 0$$

Challenge: NP-hard problem - exact methods infeasible for large graphs

Why Ant Colony Optimization?

Population-based Metaheuristic:

- **Constructive Approach:** Builds solutions from scratch (no initial solution dependency)
- **Collective Intelligence:** Multiple ants explore solution space in parallel
- **Adaptive Learning:** Pheromone-based memory guides search toward high-quality solutions
- **Balanced Exploration-Exploitation:** Pheromone trails (exploitation), heuristic information (intensification), probabilistic selection (exploration), and evaporation (forgetting)

Research Objective:

- Systematic hyperparameter optimization using Optuna
- Comparative analysis: ACO vs Greedy vs Tabu Search
- Illuminate trade-offs between constructive and improvement-based approaches

ACO Mechanism Overview

Key Components:

1. Pheromone Matrix $\tau[v][c]$
2. Heuristic Information $\eta[c]$
3. Probabilistic Selection
4. Pheromone Update

Probability Formula:

$$P(c) = \frac{[\tau[v][c]]^\alpha \cdot [\eta[c]]^\beta}{\sum_{c'} [\tau[v][c']]^\alpha \cdot [\eta[c']]^\beta}$$

Parameters:

- α: Pheromone importance
- β: Heuristic importance
- ρ: Evaporation rate
- m: Number of ants
- Q: Pheromone deposit

Parallel Execution:

- m ants per iteration
- Multi-threading

ACO Algorithm Pseudocode - Main Loop

Initialize: pheromone matrix $\tau[v][c] \leftarrow \tau_0$

FOR iteration $=$ 1 to max_iterations:

solutions $\leftarrow$ []

FOR ant $=$ 1 to m: *// Parallel execution*

solution $\leftarrow$ ConstructSolution()

solutions.append(solution)

best_solution $\leftarrow$ argmin(solutions) *// Iteration best*

Evaporate: $\tau[v][c] \leftarrow (1-\rho) \cdot \tau[v][c]$ for all v, c

Deposit: $\tau[v][c] \leftarrow \tau[v][c] + \frac{Q}{\text{num\_colors}}$ for (v, c) in best_solution

IF no improvement for patience $\times$ max_iterations:

break *// Early stopping*

RETURN best solution found

ACO Algorithm - Ant Solution Construction

ConstructSolution(starting_node):

solution $\leftarrow \{\}$

vertices $\leftarrow$ [starting_node] + shuffle(V $\setminus$ {starting_node}) *// Assigned start, then random*

FOR each v in vertices: *// Sequential vertex coloring*

valid_colors $\leftarrow \{c$: no neighbor of v uses $c\}$

IF valid_colors is empty:

valid_colors $\leftarrow$ {new color}

// Heuristic: favor used colors (reuse)

$\eta[c] \leftarrow$ 2.0 if c already used, else 1.0

// Probabilistic selection based on pheromone and heuristic

$$P(c) \leftarrow \frac{[\tau[v][c]]^{\alpha} \cdot [\eta[c]]^{\beta}}{\sum_{c' \in valid\_colors} [\tau[v][c']]^{\alpha} \cdot [\eta[c']]^{\beta}}$$

solution$[v] \leftarrow$ select c with probability $P(c)$

RETURN best solution found

Solution Representation & Constraint Handling

Solution Representation

Dictionary mapping: Solution $= \{v_1 : c_1, v_2 : c_2, \ldots, v_n : c_n\}$

Constraint Handling: Preserving Strategy

- Only valid colors considered (no conflicts with neighbors)
- Dynamic pheromone matrix expansion when needed
- **Guarantees:** All solutions are feasible ($F(C) = 0$)

Population Generation

- New population generated at each iteration
- Assigned starting nodes (distributed for coverage)
- Shuffled visitation order for remaining nodes

Diversification & Intensification

Intensification:

- Pheromone reinforcement from iteration-best
- Alpha (α) controls pheromone weight
- Heuristic preference ($\eta[c] = 2.0$) for reusing colors

Diversification:

- Pheromone evaporation $(1 - \rho)$
- Beta (β) controls heuristic weight
- Probabilistic color selection
- Random node ordering
- Parallel construction (m ants)

Balance: Adaptive mechanism between exploration and exploitation

Hyperparameter Tuning with Optuna

Optimization Setup

- **Framework:** Optuna TPE (Tree-structured Parzen Estimator)
- **Trials:** 40 trials
- **Parallel Workers:** 6 (n_jobs=6)
- **Duration:** 46 hours wall-clock (225 hours computational)
- **Tuning Dataset:** gc_500_9 (500 vertices, 90% density)
- **Objective:** Minimize total colors

Tuning Results

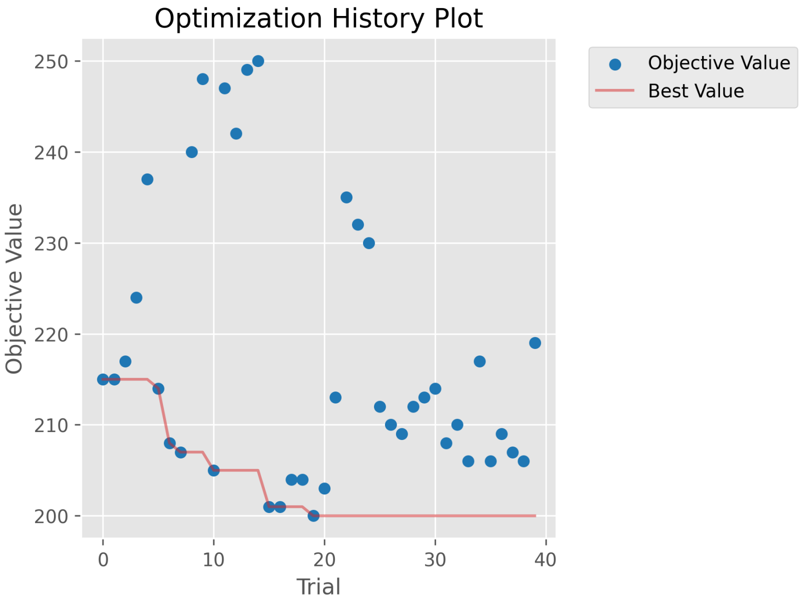
- Started at **215 colors**
- Best trial (Trial 19): **200 colors**
- **Improvement:** 15-color reduction (7%)

Optimal Parameters

Parameter	Search Range	Optimal Value
Iterations (T)	[200, 500]	261
Alpha (α)	[0.5, 2.0]	1.536
Beta (β)	[1.0, 10.0]	**5.966**
Rho (ρ)	[0.01, 0.5]	**0.097**
Ant Count (m)	[20, 100]	82
Pheromone Deposit (Q)	[0.1, 5.0]	1.299
Patience Ratio	[0.3, 0.8]	0.577

Key Findings: High β (strong heuristic guidance) and low ρ (slow evaporation) $\rightarrow$ Intensification over Diversification for GCP

Optimization History



Optimization Timeline

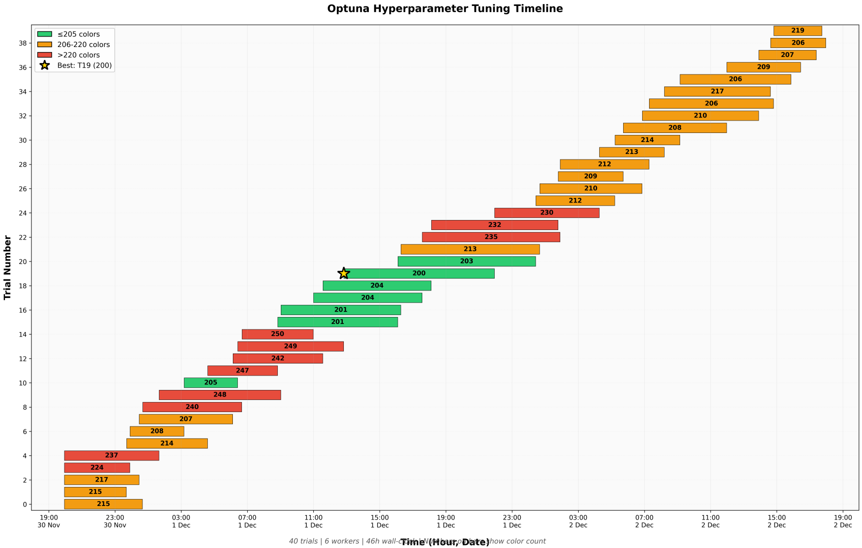


Figure: Parallel execution timeline showing 6 workers running 40 trials over 46

Parameter Analysis

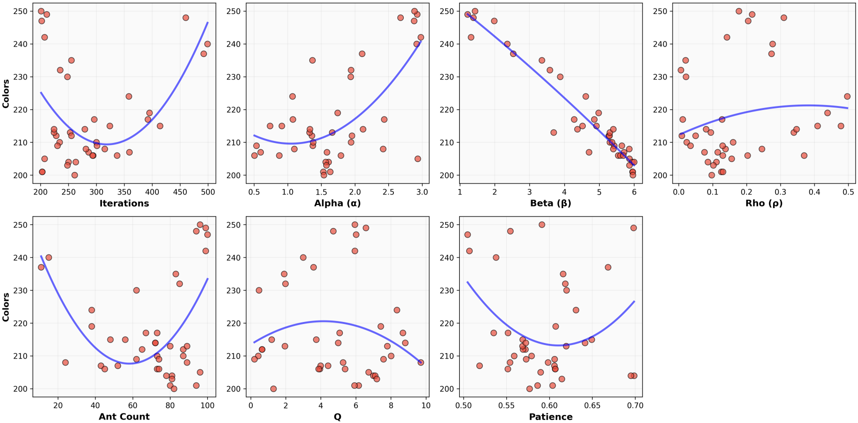


Figure: Parameter slice plots - Beta and iterations show clear trends

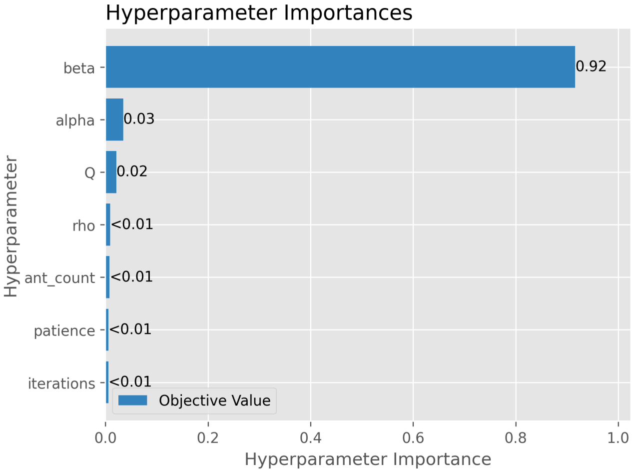
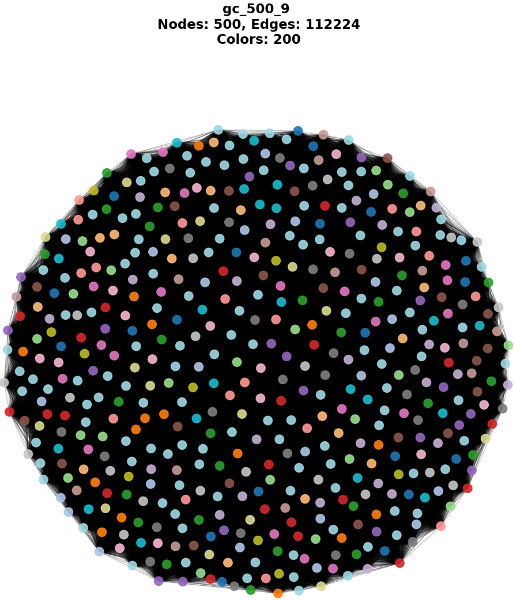


Figure: Iterations and Beta dominate performance; Rho has minimal impact

Best Trial Visualization



Benchmark Datasets

Instance	Vertices	Edges	Density	BKS
dsjc250.5	250	31,336	50.3%	28
dsjc500.9	500	224,874	90.0%	126
dsjc1000.5	1000	249,826	50.0%	85

Test Protocol:

- DIMACS benchmark instances
- 3 independent runs per instance
- Statistical robustness assessment

Comparative Performance Analysis

Instance	BKS	Algorithm	Best	Avg.	Std.	Time (s)	Dev. (%)
dsjc250.5	28	Greedy	42	42.0	0.00	0.04	50.0
		Tabu Search	34	34.7	0.58	925.7	21.4
		ACO	**55**	**55.7**	**0.58**	**1134.4**	**96.4**
dsjc500.9	126	Greedy	174	174.0	0.00	0.24	38.1
		Tabu Search	164	166.0	2.00	6730.9	30.2
		ACO	**199**	**201.3**	**2.52**	**5853.5**	**57.9**
dsjc1000.5	85	Greedy	125	125.0	0.00	0.73	47.1
		Tabu Search	124	124.7	0.58	10026.2	45.9
		ACO	**226**	**227.0**	**1.00**	**13669.6**	**165.9**

Key Finding: ACO underperforms both Greedy and Tabu Search

Deviation: 58-167% vs BKS (TS: 21-46%, Greedy: 38-50%)

Best Colors Comparison

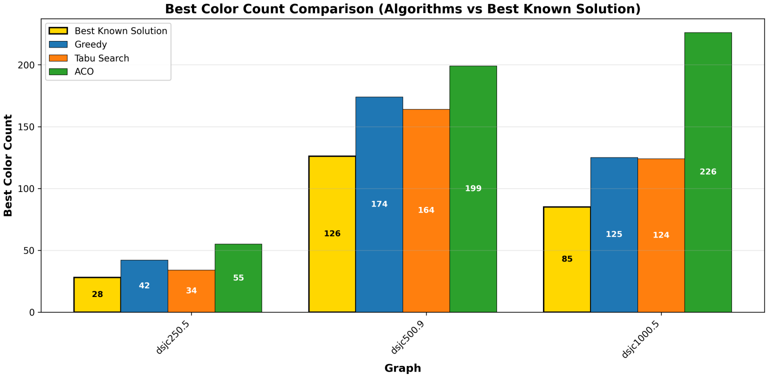


Figure: ACO produces 58-166% more colors than BKS

Average Colors Comparison

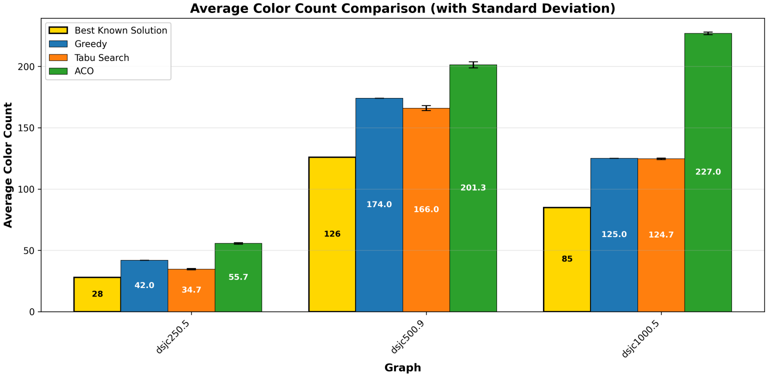


Figure: Performance gap widens with graph size: 61% (250v) → 82% (1000v)

Execution Time Comparison

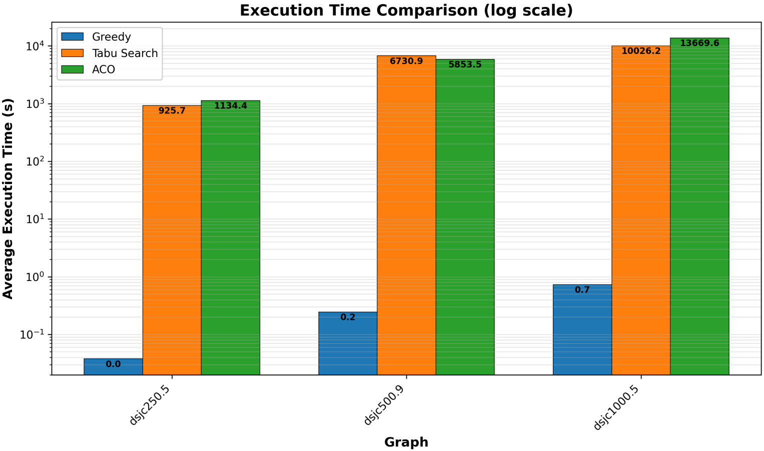


Figure: ACO time comparable to TS despite 82 parallel ants

Algorithm Comparison Summary

Greedy (Baseline)

Strength: Extremely fast (less than 1 second)
Weakness: Moderate quality (38-50% deviation)
Use Case: Rapid prototyping, time-critical applications

Tabu Search (Assignment 2)

Strength: Best solution quality (21-46% deviation)
Weakness: High computational cost (926-10,026 seconds)
Use Case: Quality-critical applications with time budget

ACO (This Work)

Strength: Consistent, parallel construction, no initial dependency
Weakness: Poorest quality (58-167% deviation), comparable time to TS
Finding: Constructive approach struggles with color minimization

Why ACO Underperforms

1. **Constructive Nature:**
   - Builds from scratch vs TS starts with 40 colors and improves to 35
   - Must discover good assignments purely through pheromone learning
2. **Local Construction:**
   - Ants make locally optimal decisions without global view
   - TS has global view of all conflicts
3. **Pheromone Sparsity:**
   - Large color sets (50-200 colors) fragment pheromone signals
   - TS's tabu list prevents specific moves without fragmentation

Scalability Issue: Performance degradation increases with graph size (96% deviation on 250v $\rightarrow$ 167% on 1000v)

Key Insights

Quality-Time Trade-off

- **Greedy:** Fast (less than 1 second) but moderate quality (38-50% deviation)
- **Tabu Search:** Superior quality (21-46% deviation), high cost (926-10,026s)
- **ACO:** Poor quality (58-167% deviation) despite comparable cost to TS

Paradigm Suitability

- Problem structure determines paradigm suitability, not algorithmic category
- Improvement-based methods benefit from feasible initialization
- Effectiveness depends on problem-specific heuristics

Future Work

Hybrid approaches, degree-based heuristics, alternative pheromone models tailored to graph coloring constraints

Thank You!

Questions?