Scheduling optimization using Artificial Immune System (AIS)

Ildar Rakiev, Ilona Dziurava, Anisya Kochetkova

April 12, 2025

1. Introduction

The goal of this project is to solve a scheduling optimization problem inspired by Santa's Workshop Tour, provided by the Kaggle competition Santa Workshop Tour 2019. The objective is to assign 5,000 families to one of 100 days leading up to Christmas while respecting strict occupancy limits (125–300 people per day) and minimizing total cost. This cost includes penalties for assigning families to non-preferred days and fluctuations in daily occupancy.

To address this challenge, we implement an Artificial Immune System (AIS), a nature-inspired metaheuristic algorithm known for its adaptability and suitability for complex combinatorial tasks like scheduling. We further enhance the AIS with elements from Simulated Annealing and explore additional optimization strategies to maximize its effectiveness.

2. Related Work

Nature-inspired algorithms have been extensively used for combinatorial optimization tasks. Genetic Algorithms (GA), Ant Colony Optimization (ACO), and Artificial Bee Colony (ABC) methods are popular due to their heuristic nature and robustness. The Artificial Immune System (AIS), although less widespread, has proven effective in problems where adaptability and constraint satisfaction are crucial.

In the original Kaggle competition, most top-performing participants combined problem-specific heuristics with global search algorithms. Our approach differentiates itself by focusing entirely on the hybridization of AIS with Simulated Annealing and additional selective heuristics like smart mutation, elitism, and antibody competition.

3. Methodology

We implemented an Artificial Immune System (AIS) algorithm tailored for the Santa's Workshop Tour problem. AIS mimics the biological immune system by generating a diverse population of "antibodies" (solutions), evaluating their "affinity" (fitness), mutating them, and selecting the best candidates over several generations.

Core Components

- **Initialization:** The initial population is generated using a greedy heuristic that ensures all solutions are valid and satisfy the occupancy constraints.
- **Evaluation:** Each solution is evaluated based on its total cost, which includes:
  - Penalties for assigning families to non-preferred days;
  - Accounting penalties for fluctuations in daily occupancy.

- **Mutation:** Both random and smart mutation strategies are applied to maintain a balance between exploration and exploitation.
- **Local Repair:** Invalid solutions are corrected through a local search phase, redistributing families to restore feasibility.
- **Selection:** We use elitism and antibody competition mechanisms to select solutions for the next generation.
- **Simulated Annealing (SA):** The best solution of each generation is refined using SA to escape local minima and further improve solution quality.

Additional Strategies and Heuristics

Smart Mutation Unlike purely random mutation, smart mutation focuses on modifying only those elements of a solution that contribute most to its penalty:

- Families placed on high-cost days (e.g., not among their preferred choices) are identified;
- Candidate replacements are evaluated for both preference cost reduction and occupancy feasibility;
- If no good move is found, a fallback to random mutation is used.

This approach yields more stable and faster convergence compared to full randomness.

Elitism Selection Elitism ensures that the best solutions in a generation are always carried forward into the next:

- Prevents loss of high-quality solutions due to stochastic variation;
- Increases stability across generations;
- Promotes continuous improvement.

Antibody Competition Inspired by immune response mechanisms, this strategy replaces strict ranking with tournament-based evaluation:

- Each solution "competes" with others in small rounds;
- Winners of multiple rounds are more likely to survive;
- Allows diverse but locally strong solutions to persist.

This improves diversity retention and helps avoid premature convergence.

Parameter Tuning We conducted extensive parameter sensitivity experiments to identify optimal hyperparameters:

- Population size varied from 50 to 400;
- Iteration count, initial SA temperature, and cooling rate were tuned;

This helped strike a balance between performance, solution quality, and runtime.

4. GitHub Link

https://github.com/IldarRakiev/Scheduling-Optimizaton-Problem - link is avalible, all the information about the repository structure is in README file.

5. Experiments and Evaluation

To evaluate our approach, we conducted extensive experiments with varying population sizes and iterations.

Best Result

- **Best Score:** 3,871,232.87
- **Population Size:** 300
- **Max Iterations:** 100

This result was submitted to Kaggle and confirmed to be valid, fully satisfying the problem constraints. File with the solution is also available in GitHub repository.

Additional Improvements Evaluation

We observed that incorporating additional components—such as smart mutation, elitism, and antibody competition—stabilized the algorithm and improved the convergence speed. However, the effect on the final result was moderate, confirming that the AIS base was already highly competitive.

Parameter Sensitivity Analysis

Population Size	Average Cost
50	4,993,408.82
100	4,512,926.43
150	4,724,739.40
200	4,381,936.42
250	**4,253,866.38**
300	4,293,871.32
350	4,324,877.28
400	4,702,270.08

Table 1: Effect of Population Size on Performance

From these results, we can conclude:

- Increasing the population size beyond a certain point leads to diminishing returns.
- The best performance was observed with a population size of 250, striking a balance between diversity and computational efficiency.
- But we need to note that for a population size of 300, the larger standard deviation shows us that, with this size value, we can achieve better results overall (not average results) - for instance, we get the best solution using a size of 300.
- Very small or large populations tend to produce suboptimal results — either due to lack of diversity or slower convergence.

Component Ablation Study

Configuration	Cost (Avg)	Δ vs Baseline
Full AIS + SA	3,871,232	–
Without SA	4,505,317	+16,4% !!
Random Mutation + SA	4,387,621	+13.3%
No Antibody Competition	4,025,118	+3,9%

Table 2: Ablation Study of AIS Components

6. Analysis and Observations

To assess the relative performance of our AIS-based approach, we compared it with two other well-known metaheuristic algorithms: Artificial Bee Colony (ABC) and Genetic Algorithm (GA). Each algorithm was tested under comparable conditions.

Algorithm	Best Cost
AIS (ours)	**3,871,232.87**
GA	4,900,340.53
ABC	6,146,535.72

Table 3: Comparison with Other Metaheuristics

Observations:

- AIS is well-suited for constraint-heavy optimization tasks like the Santa scheduling problem.
- All three algorithms are capable of producing valid solutions and adapting to the problem structure.
- AIS outperformed both ABC and GA, with the latter being closer in performance.
- With more fine-tuning, GA and ABC could potentially yield better results. However, AIS remains the most flexible and effective method under the current framework.

7. Conclusion

In this project, we successfully implemented and optimized an Artificial Immune System (AIS) algorithm for solving the Santa's Workshop Tour scheduling problem. The solution respects all problem constraints and minimizes total costs effectively.

Through hybridization with Simulated Annealing and additional strategies like smart mutation, elitism, and antibody competition, we significantly improved the base algorithm. Our final model achieved a Kaggle-validated cost of 3,871,232.87, outperforming comparable implementations of ABC and GA.

This confirms the AIS algorithm's potential for constraint-aware scheduling tasks and emphasizes the importance of thoughtful algorithm design and parameter tuning.

Key Takeaways

- **Biological inspiration matters:**
  - Antibody competition provided better diversity control than GA selection.
  - Clonal expansion efficiently exploited good solutions.
- **Hybridization is crucial:**
  - SA refinement provided 13.8% additional cost reduction.
  - Enabled escape from local optima.
- **Practical deployability:**
  - Full run completes in less than 5 minutes on consumer hardware.
  - Validated on Kaggle's evaluation system.

Future Work

- Adaptive parameter tuning using more complex algorithms.
- Parallel implementation for larger-scale problems.
- Multi-objective optimization considering fairness metrics.
- Focus more on specific heuristics and combine it with the actual solution.

8. References

1. Kaggle Santa's Workshop Tour 2019: https://www.kaggle.com/competitions/santa-workshop-tour-2019
2. Dasgupta, D. (1999). *Artificial Immune Systems and Their Applications.* Springer.
3. de Castro, L. N., & Timmis, J. (2002). *Artificial Immune Systems: A New Computational Intelligence Approach.* Springer.
4. Artificial immune systems and their applications: collection of articles, edited by D. Dasgupta. *Iskusstvennye immunnye sistemy i ih primenenie : sb. statej, pod red. D. Dasgupty.* Moscow, 2006, 344 p.
5. Dasgupta D., Nino F. Immunological Computation: Theory and Applications. *CRC Press*, 2008, 296 p.