ISSN 2278-3091

Volume 13, No.3, May - June 2024

International Journal of Advanced Trends in Computer Science and Engineering

Available Online at http://www.warse.org/IJATCS1E/static/pdf/file/ijatcse031332024.pdf

https://doi.org/10.30534/ijatcse/2024/031332024

Ant-Inspired Scheduling: Integrating Constraint Satisfaction Problem with Ant Colony Optimization

Arcely Perez-Napalit

Holy Angel University, Philippines, anapalit@hau.edu.ph

Received Date : April 11, 2024 Accepted Date: May 14, 2024 Published Date: June 06, 2024

ABSTRACT

The study integrated CSP with ACO to tackle complex scheduling challenges, demonstrating the robust capabilities of this approach. The results indicate that the integrated approach not only maintained high success rates across a range of constraints but also revealed the importance of precise parameter tuning in enhancing algorithm performance. Particularly, constraints that showed variations in success rates underlined the potential for further optimization to achieve consistent and effective outcomes. The effectiveness of the optimization algorithm was evaluated by measuring its success and performance rates. This approach proved to be a robust strategy for optimizing the study's problem structure, providing valuable insights into the dynamics of algorithmic performance.

Key words : ant colony, constraints, optimization, pheromone, scheduling.

1. INTRODUCTION

In optimization, metaheuristic algorithms have been a cornerstone, offering versatile and robust solutions to complex problems; this evolution is well-documented and explores adaptability [1]. The field transitioned to evolutionary algorithms; a concept further refined. These algorithms mimic natural evolutionary processes like selection and mutation [2]. As explored in the comprehensive analysis, nature-inspired algorithms draw from a broader spectrum of natural phenomena for problem-solving [3]. A nature-inspired population-based algorithm that dynamically adjusts its population size, selects specific modification operators for everyone, and controls the sampling period of optimized systems to simplify the fitness function and balance new solution searches with fine-tuning [4]. Within this domain, population-based algorithms stand out for their effectiveness in exploring vast search spaces [5].

Ant Colony Optimization (ACO), a prime example of this approach, leverages the foraging behaviour of ants to find optimal solutions, demonstrating a robust, adaptive method for complex optimization challenges [6]. The ACO algorithm is a new evolutionary algorithm, which is gradually applied due to its easy robustness with other methods and excellent distributed computing mechanism [7]. ACO, introduced in the 1990s and part of the broader field of swarm intelligence, mimics real ant behaviors to solve combinatorial optimization problems, where artificial ants construct solutions and communicate their effectiveness like actual ants.

Nurse scheduling, the assignment of nurses to specific tasks, is complicated by differing objectives and constraints that vary by country and hospital, with resource shortages adding to this complexity. Despite its complexity, many hospitals still manage nurse scheduling manually, a tedious process that may not comply with hospital rules. Research on medical staff allocation has predominantly focused on single aspects, with limited investigation into comprehensive allocation planning or staff preferences [8]. In Emergency Medical Services (EMS) systems, efficient staff scheduling is critical as it directly impacts patient care quality and the influence of shift sequences on staff competency [9]. Nurse scheduling, assigning nurses to various hospital tasks over a specific period, faces challenges such as resource scarcity and the risk of non-compliance with nursing regulations through manual scheduling [10]. The importance of healthcare scheduling research in optimizing costs, enhancing patient flow, and ensuring effective treatment administration in hospitals highlights the need for maximal resource utilization [11]. An objective function for nurse scheduling that minimizes constraints and balances workloads proved particularly effective in small to medium-sized problems with a semi-random initialization; their method achieved significantly lower objective values, highlighting its efficiency in optimizing nurse workloads [12]. ACO enhances maintenance, repair, and overhaul scheduling. They focused on reducing scheduling time and job tardiness and adapting to frequent shop floor changes; their findings indicated that the algorithm outperformed commercial software in solution quality and identified ways to accelerate its convergence time [13]. A cloud computing task scheduling algorithm using the ant colony algorithm achieves greater efficiency in minimizing makespan, reducing costs, and balancing system load [14]. A maintenance scheduling framework using ACO aims to optimize preventive and predictive maintenance schedules. Considering multiple constraints, this approach sought to reduce maintenance time and production losses [15]. Using the ant colony algorithm in university teaching management systems enhances efficiency through automated and manual course scheduling and adapts to intelligent teaching environments [16]. Another novel maintenance scheduling framework using Ant Colony Optimization focuses on efficient preventive and predictive maintenance scheduling to minimize maintenance time and production losses while managing multiple constraints [17].

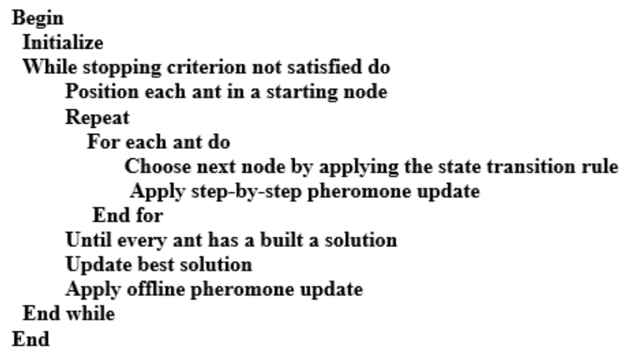
The Constraint Satisfaction Problem (CSP) provides a robust framework for modeling and solving combinatorial problems in various domains by seeking feasible solutions that adhere to specific constraints [18]. It involves finding the optimal solution from a set of feasible options within constrained optimization models, which are influenced by the problem's context aimed at achieving a particular goal [19]. Metaheuristics, derived from diverse mathematical disciplines, are crucial in tackling complex optimization problems by exploring the solution space efficiently and effectively. These algorithms excel in navigating constraints and finding feasible solutions, offering valuable strategies for a wide range of optimization issues, especially in CSPs. By exploring the dynamic relationship between decision variables, objective functions, and constraints, metaheuristic algorithms can identify optimal or near-optimal solutions while strictly complying with the problem's structural constraints.

The primary objective of this study is to effectively integrate Constraint Satisfaction Problems (CSP) with Ant Colony Optimization (ACO) to enhance staff scheduling processes, ensuring that schedules not only meet predefined constraints but are optimized. The study aims to meticulously apply and assess various parameter fine-tuning techniques to ACO, seeking to determine the optimal settings that yield the best outcomes regarding objective function values. Additionally, it will evaluate the influence of these parameter adjustments on the efficiency and effectiveness of ACO in managing the intricate demands of staff scheduling. By achieving these goals, the research provides a comprehensive understanding of how CSP-integrated ACO can be specifically tailored and refined for effective application in real-world scheduling scenarios, potentially offering valuable insights into its broader applications in organizational resource management.

2. ANT COLONY OPTIMIZATION (ACO)

The Ant Colony Optimization (ACO) concept, first introduced by Dorigo in 1992, explores artificial systems inspired by the behavior of actual ant colonies, which are utilized for solving discrete optimization problems [20]. ACO algorithm mimics actual ant behavior, using artificial pheromones for communication among artificial ants to solve problems collectively. This indirect communication allows the ants to share information and adapt their strategies based on experience, distinguishing ACO from other heuristic methods by enabling ongoing solution refinement [21]. The ACO algorithm is based on the natural foraging behavior of ants, which use pheromone trails to signal the location and quality of food sources to their colony. This process naturally optimizes the search for resources by favoring shorter, more resource-rich paths, a principle applied in the ACO to solve complex optimization problems efficiently [22]. Figure 1 presents the ACO algorithm process.

==



```
Begin
  Initialize
  While stopping criterion not satisfied do
      Position each ant in a starting node
      Repeat
        For each ant do
            Choose next node by applying the state transition rule
            Apply step-by-step pheromone update
        End for
      Until every ant has a built a solution
      Update best solution
      Apply offline pheromone update
  End while
End
```

==

Figure 1: ACO Pseudocode [23]

Ants face a decision to either go left or right, making this choice randomly. The accumulation of pheromones is more rapid on the shorter path. This variance in pheromone density between the$^{\alpha}$ paths over time guides the ants to$^{\beta}$ opt for the shorter one. During the algorithm's execution, ants initially create a range of solutions at random. These solutions are then refined by adjusting the pheromones based on the specific problem and the method of navigating the graph, with pheromones being drawn on the graph's vertices or edges. The process of moving between two nodes, i and j, is influenced by the probability associated with that edge, presented in (1). The (1) helps determine the likelihood that an ant will choose a specific path from one point to another. This decision is based on two main factors: the amount of pheromone on the path ($\tau(t)_{ij}$) and other helpful path qualities (η_{ij}), like how direct or safe the path is. The equation compares one option's pheromone level and path quality to the sum of these values for all possible paths the ant could take from its current position. This helps the ant decide on the most appealing route by looking at how well-traveled it is and how beneficial its characteristics are compared to other available paths. Before ants update the pheromone levels, which helps future ants make decisions, they look for minor improvements or better paths in the area. A local search is helpful. Updating the pheromone levels is followed, which helps reinforce good paths and diminish less useful ones over time. In (2) $\tau ij(t)+\Delta\tau ij$ (t) describes how the pheromone level on a path between two points (i and j) updates over time. At each time step, the existing pheromone amount ($\tau ij(t)$) is reduced slightly (by a factor of ($1-\rho$), where ρ is the rate at which pheromone evaporates). Then, new pheromone ($\Delta\tau_{ij}(t)$) added by ants that recently traveled that path is added to this reduced amount. This process helps to gradually fade out older trails that aren't being used much anymore, while strengthening those that are frequently traveled, guiding future ants to follow the most successful routes discovered by previous ants.

$$p_{ij}^{k} = \frac{\tau\,(t)_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{j\epsilon N^{k}(i)} \tau(t)_{ij}^{\alpha}\eta_{ij}^{\beta}} \quad (1)$$

$$\tau_{ij}(t+1) = (\,1-p).\,\tau_{ij}(t) + \,\Delta\tau_{ij}(t) \quad (2)$$

3. PROBLEM STRUCTURING

The study applied the problem structure and mathematical models [24] to address scheduling challenges effectively. Within this study, the subsequent policies for formulating a schedule within the given scenario have been identified. The mathematical formulation phase translates the constraints and objectives of the problem into mathematical terms. The model employed in the study consists of indices, data, decision variables, and penalty variables. Indices denote crucial components in scheduling, whereas data factors serve as constants. Decision variables represent the choices in scheduling that have a direct impact, and penalty variables account for the costs associated with breaching soft constraints. This formulation allows for effective analysis and optimization of scheduling in the study. The developed mathematical model optimizes scheduling by incorporating various constraints and input data values. A critical element in this model is the decision variable, which is pivotal in determining the optimal value for the objective function in a well-designed optimization problem. The constraints related to staff scheduling are as follows:

*Constraint 1***:** Each staff member works a maximum of one shift per day, expressed in (3)

Constraint 2: Senior staff are eligible for the dawn shift, which requires at least one senior staff per shift, unless it’s a Sunday, expressed in (4)

Constraint 3: Each shift must include at least one senior staff member, and no orderly staff are assigned to this shift, presented in (5).

Constraint 4: Orderly staff may be assigned to the morning shift, expressed in (6).

Constraint 5: At least one orderly staff member is required per shift, and no senior staff can be assigned to this shift, presented in (7)

Constraint 6: Staff are limited to a maximum of 22 working days per month, expressed in (8).

Constraint 7: Staff must be assigned exactly one Sunday shift, expressed in (9).

Constraint 8 : No staff member may work evening shifts on two consecutive days, presented in (10).

Constraint 9 : A maximum of seven staff members are assigned from Monday to Saturday, expressed in (11).

Constraint 10: Each day is either a working or a free day for staff, ensuring clear scheduling, expressed in (12).

$$x_{ijk} \leq 1, \forall i, k \quad (3)$$

$$\textstyle\sum_{i \in S} x_{i1k} \;\geq 1, \forall i, \forall k \;\notin D_{sun} \quad (4)$$

$$\sum_{i \in 0} x_{i1k} = 0, \forall i, \forall k \;\notin D_{sun} \quad (5)$$

$$\textstyle\sum_{i \in 0} x_{i2k} \;\geq 1, \forall i, \forall k \;\notin D_{sun} \quad (6)$$

$$\sum_{i \in S} x_{i2k} = 0, \forall i, \forall k \;\notin D_{sun} \quad (7)$$

$$\textstyle\sum_{j}\sum_{k} x_{ijk} \leq 22 \;\; \forall i \quad (8)$$

$$\textstyle\sum_{k \in D_{sun}} x_{ijk} = 1, \;\; \forall i \quad (9)$$

$$x_{i4k} + x_{i4(k+1)} \;\leq 1, \forall i, k \quad (10)$$

$$\textstyle\sum_{i}\sum_{j} x_{ijk} \;\leq 7, \;\forall k \;\in D_{sun} \quad (11)$$

$$\textstyle\sum_{i} \; x_{ijk} + \; y_{ik,} \forall i, \forall k \quad (12)$$

Given the nature of the problem in the study, it is classified as a Constraint Satisfaction Problem (CSP), which requires the state to fulfill various constraints. The goal is to identify combinations of values that meet all the specified constraints. Integrating CSP into an ACO framework necessitates adapting the algorithm to manage and adhere to these defined constraints effectively.

Integrating CSP into ACO begins by representing the CSP as a graph where nodes are decision variables, and edges are potential variable values. Ants randomly generate initial solutions that tentatively meet CSP constraints, starting with an empty solution and gradually building a complete set of variable assignments. Pheromone levels guide the search, with trails being strengthened or weakened based on solution quality to influence subsequent ant decisions. During solution construction, ants evaluate and adhere to constraints, adjusting pheromone levels accordingly if constraints are violated. Heuristic information gauges how closely a solution meets all constraints, aiding decision-making. The solutions are regularly updated based on performance, with pheromone evaporation preventing premature convergence on suboptimal solutions. A global update enhances pheromones on effective paths after all ants have constructed their solutions, and the iterative process continues, using refined pheromone trails to seek improved solutions until a satisfactory outcome is achieved or the iteration limit is reached. Figure 2 displays the pseudocode illustrating how the CSP is incorporated into the ACO process.

The study extensively explored the impact of α and βparameter settings on the ACO algorithm, with values ranging from 0.2 to 2.0 across ant populations of 50, 100, 150, and 200. Each parameter configuration was tested 20 times to validate the results statistically. This rigorous experimentation, crucial in optimization problems, helps to understand how variable manipulation affects outcomes. The combination of computational and experimental tests aimed to identify the optimal solution, employing statistical metrics such as mean, standard deviation, maximum, and minimum values to assess the algorithm's performance thoroughly. The overall effectiveness of the optimization strategy was quantified by evaluating the success and performance rates, providing a comprehensive view of how different settings influence the algorithm's ability to solve complex scheduling scenarios efficiently and demonstrating the significant role of parameter tuning in enhancing the ACO's functionality. The study utilized a computer system equipped with an Intel(R) Core™ i5-7400 CPU @ 3.00GHz, 8.00GB of RAM, and running on Windows 10 Pro operating system. This setup was chosen to guarantee the optimization model's practical implementation and effective resolution. The data utilized in this study represents the actual scheduling of the technician staff from the case study organization. This real-world data offers valuable insights into the organization's scheduling practices and operations, providing a realistic representation of the challenges and complexities faced in the field. With the actual scheduling data, the study's findings are more credible and reliable, based on real-world scenarios, and applicable to real-world situations.

Figure 2 effectively demonstrates the integration of the CSP framework with the ACO process within the study. This integration is visually represented through a schematic that highlights how elements of the CSP—variables, domains, and constraints—are incorporated into the ACO's operational flow. The figure likely details the adjustments made to the standard ACO algorithm to include CSP elements, focusing on how ants generate solutions that adhere to predefined constraints. It illustrates modifications in pheromone updates and heuristic guidance to ensure that the solutions produced seek to optimize and strictly comply with the CSP stipulations. This approach optimizes the exploration of the solution space and provides the validity of solutions under CSP constraints, enhancing the algorithm's effectiveness and applicability in complex problem-solving scenarios.

```
Initialize
    Set parameters
    Initialize pheromone levels on all paths
    Define the CSP constraints and information for each decision variables

Procedure
    for each iteration:
        for each ant:
            Initialize empty solution ( for the current ant)
            while solution not complete:
                Select variable to assign ( for the CSP order)
                Calculate probabilities for each variable
                Select a value based on the probability(value)
                    Compare the value if good for CSP constraints:
                        Assign the value to the variable ( ant's current solution)
                        Update locale pheromone
                    else:
                        Reduce pheromone ( penalty the path)
            Evaluate the completed solution(adherence to CSP)
            Compare the solution to the best solution found:
                Update best solution
        For each path ( best solution):
            Increase pheromone
        Reduce pheromone on all other paths
    Return best solution found
Print best solution
```

Figure 2: ACO Pseudocode for the given CSP structure

4. RESULTS

The analysis of running times for varying numbers of ants and parameter settings (α and β) in the ACO algorithm presented in figure 3 reveals a correlation between higher ant counts and increased computation times, particularly evident under higher β values. For example, with 200 ants, running times peak significantly at β = 0.4 and 1.2. This trend suggests that increasing emphasis on pheromone strength (α) might help reduce computation time in specific scenarios, notably at α = 2, where times are generally lowest across all ant counts. However, mid-range values of α and β often lead to longer running times, indicating that a balanced emphasis on pheromone trails and heuristic information necessitates greater computational resources due to more extensive solution space exploration. This pattern underscores the interplay between α, β, and ant counts in determining the algorithm's efficiency.

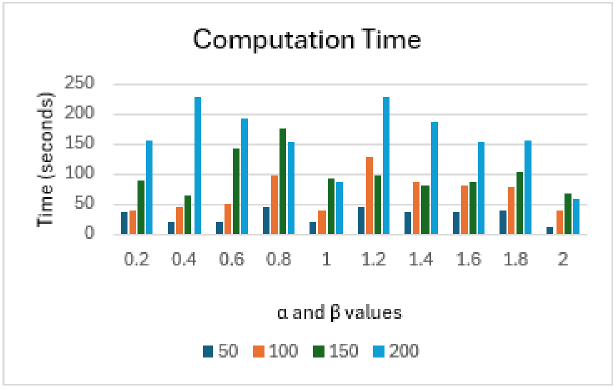


Figure 3: ACO Computation time

The data from figure 4 illustrates how the objective function values vary with different settings of α (alpha) and β (beta) parameters and other ant counts in an Ant Colony Optimization (ACO) framework. As the number of ants increases, there is a general trend toward better optimization of the objective function, especially at higher ant counts with specific parameter settings. For instance, with 200 ants, the best objective function values are observed at α = 1.6 and β = 1.4 or 1.6, achieving values as low as 0 and 1, respectively. This suggests that higher ant counts can more effectively explore and exploit the solution space, particularly when the influence of the pheromone trail strength (α) is balanced with the heuristic desirability (β). Notably, the parameter setting of α = 2 consistently shows improvements across all ant counts, achieving some of the lowest values in the dataset, indicating that a more substantial reliance on pheromone decay (higher β values) potentially leads to more optimal solutions. This pattern highlights the significant role of parameter tuning in achieving the best results in ACO, demonstrating that careful adjustment of α and β can lead to substantially different outcomes in the optimization process.

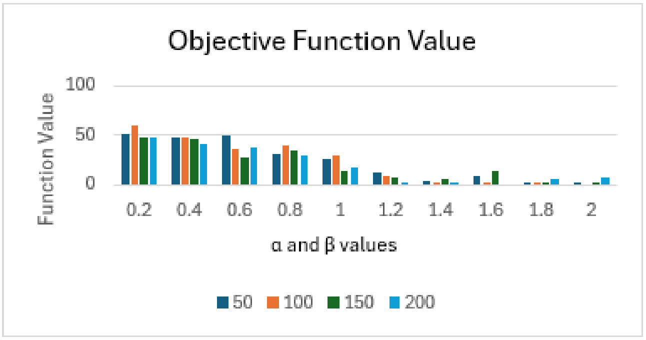


Figure 4: Best Objective Function Values

The analysis of performance and success rates across various α and β settings in the algorithm demonstrates strong effectiveness in adhering to constraints, particularly noticeable in the data from 50, 100, 150, and 200 ants presented in Tables 1 through 4. Success rates for nearly all constraints consistently hit the 100% mark under multiple parameter configurations, significantly when α and β values are set higher. This trend suggests that increasing these parameters enhances the algorithm's ability to meet specified constraints effectively.

The analysis of success rates, presented in Table 1 with 50 ants population, reveals varying performance levels across settings of α and β. C3 consistently shows high success, demonstrating robust adherence under most parameter settings, with slight reductions in a few instances. In contrast, C6 and C7 exhibit more variability, with C6 generally performing well, especially at higher levels of α, indicating sensitivity to the balance between pheromone strength and heuristic information. C7, however, presents challenges, showing lower success rates, particularly at higher α values, but also offering a glimmer of hope for improvement through parameter tuning.

Table 1: Performance and Success Rate of Constraints for 50 ant's population

α and β	Success Rate							Mean *P. Rate
	C1	C2	C3	C4	C5	C6	C7	
0.2	100	100	100	100	100	96.88	90.63	98.21
0.4	100	100	100	100	100	96.88	96.35	99.03
0.6	100	100	97	100	100	96.88	93.75	98.23
0.8	100	100	100	100	100	96.25	92.71	98.42
1.0	100	100	100	100	100	97.50	93.75	98.75
1.2	100	100	100	100	100	99.38	93.75	99.02
1.4	100	100	99	100	100	98.13	93.23	98.62
1.6	100	100	100	100	100	97.50	95.83	99.05
1.8	100	100	100	100	100	99.38	94.27	99.09
2.0	100	100	99	100	100	99.38	91.67	98.58

*Performance Rate

The results illustrated in Table 2 with 100 ants population reflect varied performance across different settings of α and β. C3 mostly maintains high success, indicative of the algorithm's strong capability in handling this constraint under a range of parameter settings. However, there is a notable dip at the highest α value. Meanwhile, C6 generally shows high compliance but with some fluctuations in success that could indicate areas sensitive to specific parameter adjustments. The most variability is observed with C7, where the success rate tends to decrease under certain conditions, suggesting that this constraint is particularly challenging for the algorithm, especially at higher α settings.

Table 2: Performance and Success Rate of Constraints for 100 ant's population

α and β	Success Rate							Mean *P. Rate
	C1	C2	C3	C4	C5	C6	C7	
0.2	100	100	100	100	100	98.75	92.71	98.78
0.4	100	100	99	100	100	98.75	93.75	98.79
0.6	100	100	100	100	100	95.63	92.71	98.33
0.8	100	100	100	100	100	98.13	89.58	98.24
1.0	100	100	99	100	100	98.13	93.23	98.62
1.2	100	100	100	100	100	96.88	91.67	98.36
1.4	100	100	100	100	100	100	91.67	98.81
1.6	100	100	100	100	100	99.38	93.23	98.94
1.8	100	100	100	100	100	99.38	90.10	98.50
2.0	100	100	96	100	100	98.13	92.19	98.04

*Performance Rate

Table 3 presents the results for 150 ants population across various α and β settings, illustrating strong and consistent performance, with all constraints achieving near-perfect or perfect success rates in most configurations. The algorithm performs exceptionally well for C3, consistently maintaining a 100% success rate, except at the highest α setting, where it slightly dips. While C6 is also generally high, the success rate shows some variability, reaching perfect compliance at the highest α level, suggesting that this constraint benefits from increased pheromone influence. Constraint C7 experiences slight variations in success across different settings, with the lowest performance observed at lower β values, indicating a potential area for tuning to consistently enhance the algorithm's ability to meet this more sensitive constraint.

Table 3: Performance and Success Rate of Constraints for 150 ant's population

α and β	Success Rate							Mean *P. Rate
	C1	C2	C3	C4	C5	C6	C7	
0.2	100	100	100	100	100	96.88	91.67	98.36
0.4	100	100	100	100	100	97.50	88.54	98.01
0.6	100	100	100	100	100	98.75	92.71	98.78
0.8	100	100	100	100	100	97.50	92.71	98.60
1.0	100	100	100	100	100	97.50	94.79	98.90
1.2	100	100	100	100	100	95.00	92.19	98.17
1.4	100	100	100	100	100	98.13	95.31	99.06
1.6	100	100	100	100	100	99.38	93.75	99.02
1.8	100	100	100	100	100	99.38	93.23	98.94
2.0	100	100	99	100	100	100	93.75	98.96

*Performance Rate

The success rates presented in Table 4 for 200 ants population across different α and β indicate an overall robust performance. The algorithm consistently achieves perfect or near-perfect success rates for C3, maintaining 100% in most configurations. For C6, success rates are generally high, demonstrating 100% success, suggesting optimal parameter alignment for this specific constraint. However, the variability is observed in constraint C7, particularly at higher β settings, with a notable dip in performance at the highest β value, indicating potential oversensitivity to parameter increases in this area.

Table 4: Performance and Success Rate of Constraints for 200 ant's population

α and β	**Success Rate**							**Mean *P. Rate**
	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	
0.2	100	100	100	100	100	97.50	92.19	98.53
0.4	100	100	100	100	100	95.00	94.27	98.47
0.6	100	100	99	100	100	96.25	94.79	98.58
0.8	100	100	100	100	100	98.75	93.23	98.85
1.0	100	100	100	100	100	98.75	94.27	99.00
1.2	100	100	100	100	100	99.38	94.79	99.17
1.4	100	100	100	100	100	100	90.10	98.59
1.6	100	100	100	100	100	99.38	93.75	99.02
1.8	100	100	100	100	100	98.13	92.19	98.63
2.0	100	100	100	100	100	98.75	93.75	93.93

The analysis of the optimization results, as presented in Table 5, reveals critical insights into the algorithm's performance, reflected in the objective function values and running times. The mean objective function value stands, indicating the average effectiveness of the algorithm across various runs, with a standard deviation of 6.25, which suggests a moderate variability in performance. Additionally, the mean running time, pointing to the average computational effort required by the algorithm is complemented by a standard deviation of 17.50 seconds. This latter measure indicates some fluctuations in the running times, which may be attributed to differences in problem complexity or computational efficiency across different tests. Together, these statistics provide a comprehensive overview of the algorithm's operational efficiency and effectiveness, highlighting areas of stability and aspects that might benefit from further optimization to reduce variability and improve performance consistency.

Table 5: ACO Algorithm result

Analysis	Result
Mean Objective Function Value	13.03
Std. Dev Objective Function Value	6.25
Mean Running Time	78.08
Std. Dev. Running Time	17.50

The submitting author is responsible for obtaining agreement of all coauthors and any consent required from sponsors before submitting a paper. It is the obligation of the authors to cite relevant prior work.

5. CONCLUSION

A conclusion section is not required. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

ACKNOWLEDGEMENT

The researcher would like to express gratitude to the RLLMH for their invaluable support in completing this study.

REFERENCES

1. C. Blum and A. Roli. **Metaheuristic optimization**. *Artificial Intelligence Review 2003, 21*(1), 3-44. doi:10.1023/A:1022103526203.
2. A..E Eiben and J.E. Smith.. **Introduction to Evolutionary Computing (2015)**. Springer. doi:10.1007/978-3-662-44874-8.
3. I. Fister, I. Fister, X.S. Yang, and J. Brest. **A comprehensive review of firefly algorithms.** *Swarm and Evolutionary Computation* (2013), *13*, 34-46. doi:10.1016/j.swevo.2013.06.001
4. K. Cpałka, A. Słowik, and K. Łapa. **A population-based algorithm with the selection of evaluation precision and size of the population**. *Applied Soft Computing(2022), 115*, 108154. https://doi.org/10.1016/j.asoc.2021.108154
5. A.P. Engelbrecht. **Computational Intelligence***: An Introduction*. Wiley (2007).
6. M. Dorigo, and T. Stützle, T. **Ant Colony Optimization**. MIT Press(2004).
7. R. Ge, and J. Chen, J. **Analysis of college course scheduling problem based on Ant Colony Algorithm**. *Computational Intelligence and Neuroscience*, *2022*, 1–9. https://doi.org/10.1155/2022/7918323
8. O. Chen and Z. Zeng. **Developing two heuristic algorithms with metaheuristic algorithms to improve solutions of optimization problems with soft and hard constraints: An application to nurse rostering problems**. *Applied Soft Computing(2020)*, *93*, 106336. https://doi.org/10.1016/j.asoc.2020.106336
9. H. Vermuyten, J.N. Rosa, I. Marqués, J. Beliën and A.P. Barbosa-Póvoa . **Integrated staff scheduling at a medical emergency service: An optimisation approach**. *Expert Systems With Applications(2018)*, *112*, 62–76. https://doi.org/10.1016/j.eswa.2018.06.017
10. L. Hakim, T. Bakhtiar, and Jaharuddin. **The nurse scheduling problem: a goal programming and nonlinear optimization approaches**. *IOP Conference Series (2017)*, *166*, 012024. https://doi.org/10.1088/1757-899x/166/1/012024
11. Z.A. Abdalkareem, A. Amir, M.A. Al-Betar, P. Ekhan and A.I. Hammouri.. **Healthcare scheduling in optimization context: a review**. *Health and Technology(2021)*, *11*(3), 445–469. https://doi.org/10.1007/s12553-021-00547-5

12. S. Achmad, A. Wibowo and D. Diana. **Ant colony optimization with semi random initialization for nurse rostering problem(2021).** *International Journal for Simulation and Multidisciplinary Design Optimization*, *12*, 31. https://doi.org/10.1051/smdo/2021030
13. L.V. Tran, B.H. Huynh, and H. Akhtar. **Ant Colony Optimization Algorithm for Maintenance, Repair and Overhaul scheduling optimization in the context of industries 4.0(2019)**. *Applied Sciences*, *9*(22), 4815. https://doi.org/10.3390/app9224815
14. Q. Guo. **Task scheduling based on ant colony optimization in cloud environment(2017).** *Nucleation and Atmospheric Aerosols*. https://doi.org/10.1063/1.4981635
15. A. Kaul. **An innovative maintenance scheduling framework for preventive, predictive maintenance using Ant colony optimization(2022**). In *IntechOpen eBooks*. https://doi.org/10.5772/intechopen.103094
16. W. Yi. **Research on the application of Ant colony Algorithm in University Teaching Management Service System (2022).** *MATEC Web of Conferences*, *359*, 01020. https://doi.org/10.1051/matecconf/202235901020
17. A. Kaul. **An innovative maintenance scheduling framework for preventive, predictive maintenance using Ant colony optimization(2022**). In *IntechOpen eBooks*. https://doi.org/10.5772/intechopen.103094
18. W. Song, Z. Cao, J. Zhang, C. Xu, and A. Lim. **Learning variable ordering heuristics for solving constraint satisfaction problems(2022)**. *Engineering Applications of Artificial Intelligence*, *109*, 104603. https://doi.org/10.1016/j.engappai.2021.104603
19. H. Arham. **Deterministic Modeling: Linear Optimization with Applications (2020).**
20. M. Dorigo, V. Maniezzo, and A. Colorni. **The ant system optimization by a colony of cooperating agents**. *IEEE Transactions on Systems, Man, and CyberneticsPart B*, 26(1):29–41, 1996.
21. A. Akhtar. **Evolution of Ant Colony Optimization Algorithm - A Brief Literature Review (2019)**. *ArXiv, abs/1908.08007*.
22. A. Rezvanian, S. Mehdi Vahidipour, and A. Sadollah. **An overview of ant colony optimization algorithms for dynamic optimization problems (2023)**. *Ant Colony Optimization - Recent Variants, Application and Perspectives[WorkingTitle]*. https://doi.org/ 10.5772/ intechopen.111839
23. Kumar, D Nagesh & Janga Reddy, M. (2006). Ant Colony Optimization for Multi-Purpose Reservoir Operation. Water Resources Management. 20. 10.1007/s11269-005-9012-0.
24. A.P. Napalit and M.A. Ballera. **Application of Firefly Algorithm in Scheduling**. *2021 IEEE International Conference on Computing (ICOCO)*. https://doi.org/10.1109/icoco53166.2021.9673581