Journal of Total Science
A special issue of the First Scientific Conference on Science and Technology
Volume (7), Issue (26), (Nov. 2023)
ISSN: 2518-5799

SCST
November 2023

IMPROVE DYNAMIC DELIVERY SERVICES USING ANT COLONY OPTIMIZATION ALGORITHM IN THE MODERN CITY BY USING PYTHON RAY FRAMEWORK

***Llahm Omar Faraj Ben Dalla[1], Tarik Milod Alarbi Ahmad[2]**

Software Engineering Department[1], Computer Engineering Department[2]
College of Technical Science- Sebha[1], University of Gharyan[2]

llahmomarfaraj77@gmail.com[1]& tma_7444@yahoo.com[2]

Orcid.ID: 0009-0008-7624-7567[1]

.....

.....

ABSTRACT

Dynamic delivery services are becoming increasingly important in the modern city. These services allow businesses to deliver goods and services to customers in a timely and efficient manner. However, dynamic delivery services can be challenging to optimize. This is because the environment in which these services operate is constantly changing. In this paper, we propose a new approach to optimizing dynamic delivery services. Our approach uses an ant colony optimization (ACO) algorithm. ACO is a metaheuristic algorithm that is inspired by the foraging behavior of ants. ACO has been shown to be effective for solving a variety of optimization problems. This research implemented our ACO algorithm in the Python Ray framework. Ray is a distributed computing framework that can be used to run large-scale applications. This allowed us to scale our ACO algorithm to a large number of ants. This research has evaluated ACO algorithm on a real-world dataset of delivery requests. Our results show that our ACO algorithm can significantly improve the efficiency of dynamic delivery services. This research work has several implications. First, our work demonstrates that ACO can be used to effectively optimize dynamic delivery services. Second, our work shows that the Python Ray framework can be used to run large-scale ACO applications. Third, our work provides a foundation for further research on the optimization of dynamic delivery services.

Keywords: Dynamic Delivery Services, Ant Colony Optimization Algorithm, Modern City Python, Ray Framework.

1. Introduction

in modern cities, where there are a large number of people, who, in their totality, constitute the number of different societies, and coexist in small towns, especially in the suburbs, far from the center of cities, they need delivery services, to deliver orders for food, mail, gifts, and others.[1]

The need for companies and platforms to serve the delivery of orders: where the main point of the delivery service, which includes receiving orders from customers, appointing delivery personnel, scheduling and tracking deliveries. Due to the increase in demand for delivery services, there was a need to devise ways and methods to improve delivery methods for orders, with the least time, effort and the lowest cost in fuel consumption. From

here, the need for an algorithm appeared that contributes effectively to improving the previous operations, which is Ant Colony Optimization Algorithm (ACO) component optimizes delivery routes by simulating the foraging behavior of ants. The ACO algorithm finds the shortest path between delivery locations, taking into account factors such as traffic conditions. Using - Ray Framework: This component is the distributed computing infrastructure required to run the ACO algorithm. The Ray framework provides the tools to scale process optimization across multiple nodes, which is critical for delivery services at scale.[2]

- Data sources: This component includes different data sources, such as traffic data, delivery mileage, and delivery order data, which the ACO algorithm uses to optimize delivery methods.

the proposed system architecture integrates the ACO algorithm with a delivery service platform, while taking advantage of the Python Ray framework to distribute the optimization process across multiple nodes. Delivery personnel and customer applications act as interfaces to interact with the delivery service platform.

2.ANT COLONY OPTIMIZATION (ACO) ALGORITHM

Ant Colony Optimization is one of the modern algorithms used to solve optimization problems which are animal and insect in nature. This algorithm relies on the ants' foraging behavior and finding the best path back to the nest.

The ant algorithm is one of the smart behavior-based algorithms, which relies on the basic idea of employing collective intelligence that depends on the communication of living organisms with each other to find optimal solutions to problems. This intelligence is available in the ant colony. The ants algorithm improves the overall performance of the system, by improving the path that ants take to reach food, as ants seek to shorten the distance traveled between the colony and the food location, and reduce the time it takes to reach the goal. This algorithm is based on the concept of meta-monitoring, where ants secrete certain chemicals that help them decide which path to take back to the nest, and other ants check for this chemical and head in the direction it indicates.[3]

The ant algorithm consists of several steps, namely:

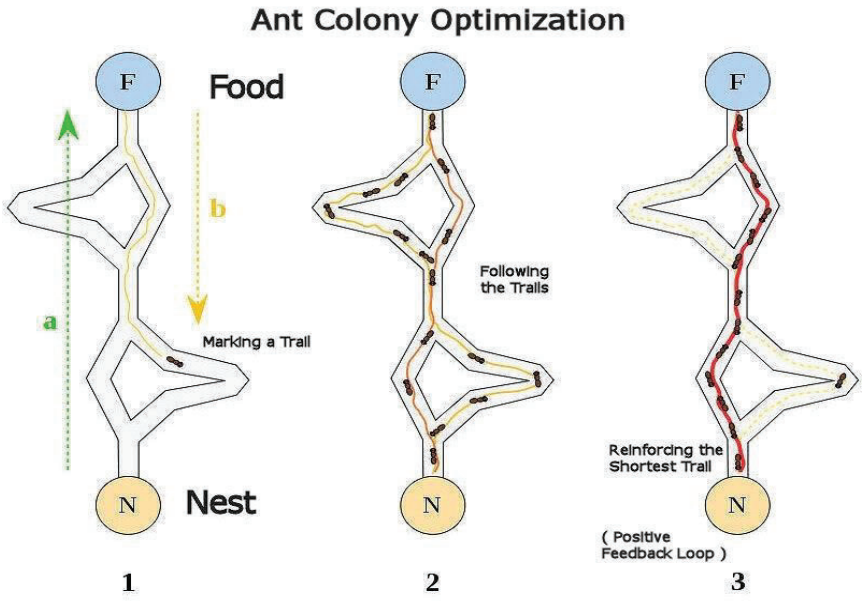
1- Creating paths: Several random paths are generated initially, and ants are selected to follow each path.
2- Updating the chemical fingerprint: The ants update the chemical fingerprint when they pass on a path, where the ants leave a chemical trace (pheromone) on the path they take, and this trace is updated whenever the ants pass on the path.
3- Choosing the optimal path: the ants check the chemical effect on the different paths, and choose the path that has a stronger effect than the pheromone.
4- Updating the top monitoring: The top monitoring is updated after choosing the optimal path, as the ants update the chemical effect on the path they take.
5- Repetition: The previous steps are repeated until the goal is reached.

The ant algorithm is used to solve many optimization problems in different fields, such as designing computer networks, designing electrical circuits, designing aircraft, cars and trains, and even in the fields of logistics, movement and transportation.[4]

From this point of view, the ant algorithm is considered one of the smart and effective algorithms in solving optimization problems, due to its ability to work in a balanced manner and distribute performance among ants,

which ultimately means improving the overall performance of the system. Because the ant colony algorithm relies on the idea of collective intelligence, which takes advantage of living organisms communicating with each other to find optimal solutions to problems. ACO algorithm shown in figure (1)

In this research, the raging ant algorithm was used to improve delivery services within the city, as a practical application, to the idea of optimization.



ACO algorithm shown in figure (1)

Python Ray Framework:

It is an open-source framework designed to facilitate the building of AI and distributed computing applications using the Python programming language.

The Python Ray Framework is used to facilitate distributed operations and deal with big data. It provides a software interface that enables users to distribute tasks and processes among multiple machines to achieve excellent performance and improve efficiency. The Ray Framework also enables computational acceleration with graphics processing and optimization of CPU and GPU utilization.[5]

The Ray Framework is based on the Task Model, where processes are divided into tasks that run independently and can be deployed on multiple machines. These tasks are executed in a distributed process automatically, and are controlled using the call and control functions available in the Python Ray Framework.[6]

The Python Ray Framework is useful for application designers working on building artificial intelligence, machine learning, and big data analytics applications, as it can be used to segment processes and distribute them across many machines to improve performance and reduce time to complete tasks. The Ray Framework can be used in

various applications such as statistical analysis, machine learning, big data analysis, software self-improvement, and parallel process orchestration.[7]

The Ray Framework provides a set of features, the most important of which are:

Distributed Processing Functions: The Ray Framework enables distributed processing functionality and splits processes into separate tasks that run independently.

Performance improvement: The Ray Framework can be used to improve performance and reduce the time required to complete complex computational tasks.

- Load Balancing: Ray Framework distributes work among many machines automatically and load balances between them.

Ray Framework is adaptable to different systems and environments, making it a flexible and easy-to-use framework for developing AI and distributed computing applications using the Python language. RAY framework shown in the Figure (2)

The algorithms and data structures used in the research to be focused on optimizing the delivery routes using the ACO algorithm, representing the problem as a graph data structure, and using appropriate data structures to store distance data and points of delivery . The Python Ray Framework is used to distribute the optimization process across multiple nodes, making it suitable for large-scale delivery services.

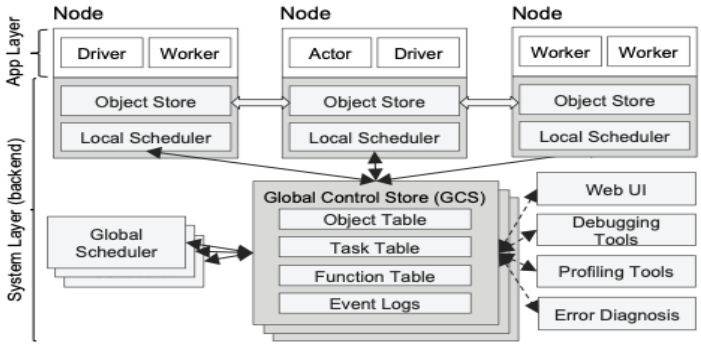
In this research, the RAY framework was used to take the storage of delivery services points, estimate the distances within the city and send them to the system for storage and analysis,

until they are improved using the ant colony algorithm.[8]

Figure(2) RAY framework

GUI Application:

First, when running an application, improving delivery services in modern cities, using the Python language, the main interface of the program appears to us, Figure (3)



Figure(2) RAY framework

GUI Application:

First, when running an application, improving delivery services in modern cities, using the Python language, the main interface of the program appears to us, Figure (3)

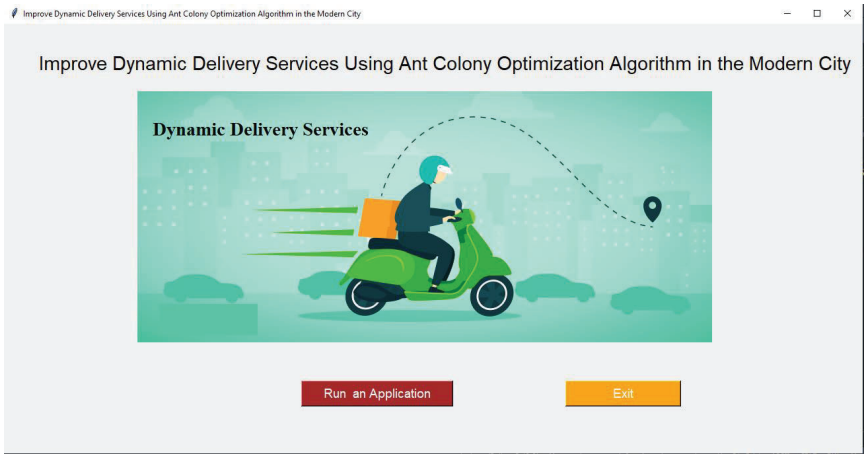


Figure (3) GUI Application

When the Run button is pressed, the RAY framework is loaded, it starts the distributed calculation of the connection points, and the process of computing routes for the connection starts in the background. figure (4)

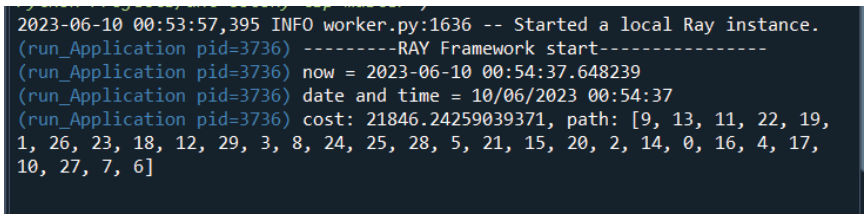


Figure (4) Ray frame work calculate path cost And best track

The system presents the proposed set of paths according to the following figure (5)

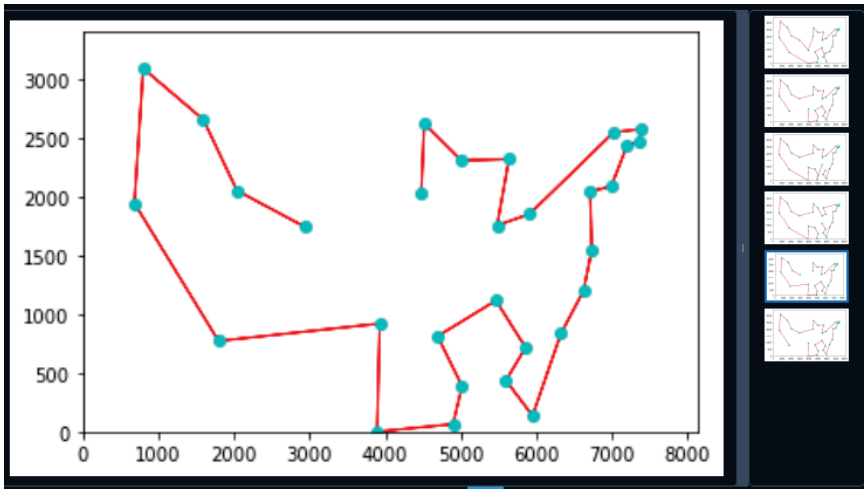
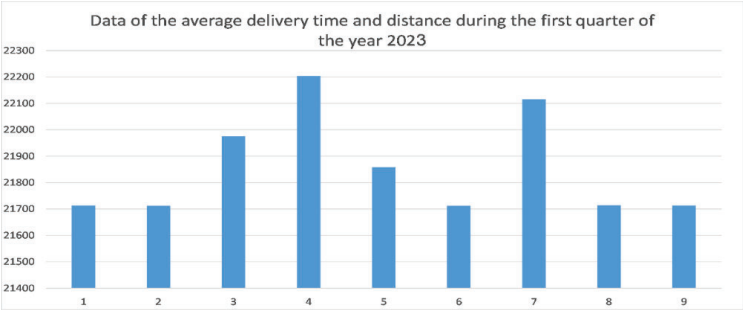


Figure (5) the best track for Delivery services from application

From the paths proposed by the system, the delivery service employee chooses the appropriate path, as the system is built on the architecture of the distributed system, which helps in understanding and analyzing the size of the network, and the closest delivery paths to the delivery service employee. The following figure shows most of the paths drawn by the system Reducing time, effort and fuel consumption for the delivery service employee.

3.EVALUATION

After implementing and testing the proposed system for Improve Dynamic Delivery Services Using Ant Colony Optimization Algorithm in the Modern City by Using Python Ray Framework, and recording distance data and delivery routes within the city during the first quarter of the year 2023, the following results appeared in Figure (6):



3Figure (6) Data of the average delivery time and distance during the first quarter of the year 2023

During the first quarter of the year 2023, the performance of the proposed system was evaluated, and the data showed that it is from the delivery service performance, using the ACO algorithm and the applied Python Ray framework. There has been an improvement in the results, and an increase in the speed of delivery of orders to

customers, and this was evident in the short delivery time, the decrease in the cost of delivery, and the accuracy of delivery before and after the implementation of the ACO algorithm. Figure (7)

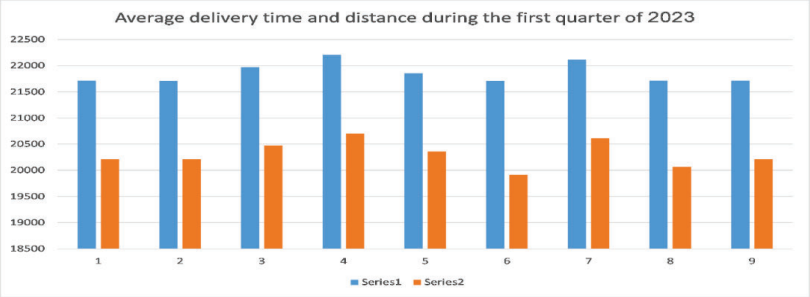


Figure (7)Average delivery time and distance during the first quarter of 2023

4.SCALABILITY AND RELIABILITY

The following scalability and reliability considerations are likely to be addressed:

Scalability: The delivery service platform must be designed to handle large amounts of data and user requests. The Ray Framework is used to provide scalability by distributing the optimization process across multiple nodes. The system must be designed to accommodate the growing number of delivery personnel and customers.

- **Reliability**: The delivery service platform must be reliable, ensuring that delivery orders are processed accurately and efficiently. The system must be designed to handle errors and failures, such as network downtime or server downtime. The ACO algorithm must be designed to handle unexpected situations such as changes in traffic conditions. Street closures, traffic problems such as road accidents, overcrowding due to festivals, sporting events, holidays and social events.

- **Fault tolerance:** The system must be designed to handle errors and recover from failures. This may include implementing troubleshooting techniques such as error logs, isolating fault areas from the system, and backup and recovery.

- **Load balancing:** The system should be designed to balance the load through multiple load distribution techniques. This ensures that no single node is not overloaded, leading to performance degradation or system failure, and out of service.

-**Security:** The delivery service platform must be designed to ensure the security of customer data and prevent unauthorized access. This may include implementing access control mechanisms, data encryption, and secure communication protocols.

Scalability and reliability considerations in this research are likely to focus on ensuring that the delivery service platform is scalable, reliable, fault-tolerant, and secure. The Ray Framework is used to provide scalability, while fault tolerance and load balancing techniques are used to ensure reliability, fault handling, and isolating the area where the fault occurred so that the entire system is not affected. Security measures must be implemented to ensure the security of customer data and to prevent unauthorized access.

5.TRADE-OFF AND LIMITATIONS

Trade-offs between optimization and real-time delivery: The ACO algorithm may optimize delivery methods based on various factors, but this may take some time to compute. This trade-off must be considered to ensure that delivery personnel can meet delivery times while continuing to follow the improved routes.

Limitations of the ACO Algorithm: The ACO algorithm may not always provide the best delivery method, especially if delivery locations change frequently. The algorithm may also be limited by the accuracy of the traffic data, mileage and routes used by delivery personnel.

Limitations of the Python Ray framework: The Python Ray framework, in addition to providing scalability, may lead to increased overheads, which may affect the performance of the delivery service.[9]

Limitations of the scope of the search: The search may be limited to a specific area, and the results may not be generalizable to other areas with different traffic conditions and different delivery requirements, especially in times of events, holidays and seasons of the year.

Implementation Limitations: The successful implementation of a delivery service platform may depend on several factors, including the availability of computing resources, the experience of the implementation team, the compatibility of the system with other external systems, the city's infrastructure and communication speed, and the capacity of the city's logistics sites.

The trade-offs and limitations of the paper may focus on trade-offs between optimization and real-time delivery, limitations of the ACO algorithm, the Ray framework, research scope limitations, and implementation limitations. These trade-offs and limitations must be carefully considered to ensure an effective and efficient delivery service platform implementation and future development.[10]

6.FUTURE WORK

The following future work may be considered:

Further improvement of the ACO algorithm: The ACO algorithm can be further improved to consider additional factors such as: adding ratings to delivery recipients, delivery times, and delivery costs. This may lead to improved delivery service performance and customer satisfaction. Integration with other optimization algorithms: The ACO algorithm can be combined with other optimization algorithms such as the genetic algorithm to provide a more comprehensive optimization solution for delivery service. System integration with real-time updates of traffic, congestion and weather data: The delivery service platform can be enhanced by integrating real-time updates of weather and traffic data within the city. This may improve the accuracy of the ACO algorithm and lead to more optimal delivery methods. Implement predictive analytics: Predictive analytics can be implemented to forecast delivery demand and improve delivery schedules. This may improve delivery service efficiency and reduce delivery costs and fuel consumption. System support by integration with other e-commerce platforms: The delivery service platform may be integrated with other e-commerce platforms such as online marketplaces or retail websites. This could increase the delivery service's reach and customer base. Hence future work for this research should focus on improving the ACO algorithm, integration with other optimization algorithms, real-time updates

of traffic and weather data, implementation of predictive analytics, and integration with other e-commerce platforms. These improvements may improve delivery service performance, scalability, and efficiency.

7.CONCLUSION

In conclusion, this paper proposes the use of Ant Colony Optimization (ACO) algorithm and Ray Framework to improve dynamic delivery services in the modern city. Where the ACO algorithm was used to optimize the delivery methods, taking into account conditions such as traffic, congestion on the streets in the city, and other factors, while the Ray Framework was used to provide scalability by applying distributed systems, in order to distribute the optimization process across multiple routes and suggest different paths For fast delivery and delivery of orders. The paper also discussed the evaluation methods, scalability and reliability considerations, limitations, and future work of the proposed delivery service platform.

The actual application of the proposed delivery service platform may have the potential to improve the efficiency, accuracy and cost-effectiveness of delivery services in the modern city. The ACO algorithm provides a reliable and efficient way to optimize delivery methods, while the Ray Framework allows the optimization process to scale seamlessly as the service grows as it depends on the distributed systems at work. The evaluation methods and scalability and reliability considerations provide a framework for ensuring that the delivery service platform is efficient, reliable and secure.

In view of the above, this paper makes a valuable contribution to the field of dynamic delivery services, and presents a promising solution to improve delivery services in the modern city. With the potential to improve, the proposed delivery service platform, and expand it to meet the evolving needs of the delivery services industry in the future.

REFERENCES

1. Baptiste, P.; Flamini, M.; Sourd, F. Lagrangian bounds for just-in-time job shop scheduling. *Comput. Oper. Res.* 2008, *35*, 906–915
2. Lawler, E.L.; Lenstra, J.K.; Rinnooy Kan, A.H.G.; Shmoys, D.B. *Sequencing and Scheduling: Algorithms and Complexity*; Technische Universiteit Eindhoven: Eindhoven, The Netherlands, 2009
3. Brucker, P.; Knust, S. *Complex Scheduling*; Springer: Berlin, Germany, 2012.
4. Xie, J.; Gao, L.; Peng, K.; Li, X.; Li, H. Review on flexible job shop scheduling. *IET Collab. Intell. Manuf.* 2019, *1*, 67–77.
5. Leusin, M.E.; Frazzon, E.M.; Uriona Maldonado, M.; Kück, M.; Freitag, M. Solving the Job-Shop Scheduling Problem in the Industry 4.0 Era. *Technologies* 2018, *6*, 107.
6. Xiong, H.; Danni, S.S.; Hu, R.J. A survey of job shop scheduling problem: The types and models. *Comput. Oper. Res.* 2022, *142*, 105731.
7. Gonçalves, J.F.; de Magalhães Mendes, J.J.; Resende, M.G. A hybrid genetic algorithm for the job shop scheduling problem. *Eur. J. Oper. Res.* 2005, *167*, 77–95.

8. Gromicho, J.A.S.; Hoorn, J.J.; Timmer, G.T. Exponentially better than brute force: Solving the job-shop scheduling problem optimally by dynamic programming. *Comput. Oper. Res. Arch.* 2012, *39*, 2968–2977.

9. Çaliş, B.; Bulkan, S. A research survey: Review of AI solution strategies of job shop scheduling problem. *J. Intell. Manuf.* 2015, *26*, 961–973.

10. Matrenin, P.V.; Manusov, V.Z. The cyclic job-shop scheduling problem: The new subclass of the job-shop problem and applying the simulated annealing to solve it. In Proceedings of the IEEE 2nd International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), Chelyabinsk, Russia, 19–20 May 2022.

11. Brucker, P.; Jurisch, B.; Sievers, B. A branch and bound algorithm for job shop scheduling problem. *Discret. Appl. Math.* 1994, *49*, 107–127.