Hybrid Metaheuristics for Last-mile Delivery Scheduling and Rescheduling

[redacted]

***Abstract*—As the delivery stage with the highest user perception and the highest cost ratio, last-mile delivery is the most correlated with the overall service quality of the logistic companies. As their performance may be affected by multiple factors, solving last-mile delivery scheduling and rescheduling problems requires high-performance algorithms. Thus, we propose and develop two hybrid metaheuristics to optimize such a problem, as well as a rescheduling algorithm. As an evaluation result, the K-means hybrid version of the Ant Colony Optimization algorithm achieves the highest cost reduction capability and convergence speed.**

***Index Terms*—multi-criteria optimization, metaheuristic, algorithm, logistic**

I. Introduction

In recent years, fulfillment networks have been a fast-growing industry. Online retail market share only contributes 18% in 2022 [1]. This indicates there is still huge potential for the online retail market and the improvement of associated last-mile delivery. Last-mile delivery, which is the stage that goods move from distribution centers to customers' doorsteps, is the most important customer engagement and is a critical stage in the entire logistics chain for logistic enterprises to improve customer satisfaction. The purpose of this paper is to provide several optimized algorithms to help e-commerce enterprises achieve high-quality, fast delivery and visible on-road monitoring among operators, shippers and customers. On the other hand, rising package costs and energy costs caused by the pandemic and inflation forced logistic shipping companies to need to further reduce costs. The last-mile delivery expenses account for a large proportion of the logistics and operation cost due to more staffing required such as dedicated drivers, central operators at warehouses, and the scattered transport capacity (different fleets, vans and line hauls).

In terms of the environmental cost, gas consumption and the resulting additional carbon emissions also need to be optimized through shorter travel distances.

During the last mile delivery workflow, stop arrival time scheduling and routing assignment tasks are big challenges. Customers often want to have a preferred delivery time window for secure deliveries. The point-to-point movement is not sustainable and profitable and cannot fulfill the requirements of clients and enterprises. In particular for customers that require a preferred time slot for delivery and pickup. Therefore, we propose several bio-inspired design algorithms targeting last-mile delivery scheduling and rescheduling and compare their execution time and cost output. This problem can be characterized as a vehicle routing problem (VRP) with many more stops and additional constraints, such as time windows and vehicle capacity.

The routing solution aims to (i) minimize the required number of delivery vehicles given a fixed amount of packages capacity (ii) minimize overall travelling distance (traffic time not included) (iii) meet the timing requirements of the clients (soft constraints), and (iv) avoid unnecessary idling. Regarding the approach to rescheduling problems, we propose to rerun the existing pilot run routing algorithm. Operators at the delivery station can re-run the route planning algorithm to fetch the latest updated routing plans. This way simplifies the operator's rescheduling work and reduces the computation time from calculating the routing algorithms from scratch.

The challenge of solving this problem includes (i) Scope: The algorithm needs to avoid local optima and have a wide field of view to find the global optimal solution. (ii) Uncertainty: The algorithm has a very large degree of freedom, which implies that finding a more efficient method is required. The final solution is heavily influenced by the initial solution (iii) Constraints: As a combinatorial optimization problem, additional constraints complicate mathematical modelling and could impact the results.

In this project similar to the travelling salesman problem, the proposal is to utilize a traditional clustering algorithm, which is K-means Clustering, to find the optimized and feasible initial solution and then feed the initial solution with several metaheuristics, which are Genetic algorithm (GA) and Ant Colony Optimization algorithm (ACO) to find the global optimal solution to route assignment and scheduling problem. We formulate the problem as a VRP and use a parcel delivery dataset sampled from the 2021 Amazon Last Mile Routing Research Challenge Dataset to train and verify our algorithm. The dataset includes last-mile delivery routes from one distribution station in Los Angeles, as well as parcel and stops information.

The structure of this paper includes related research work, proposed formulation and modelling solutions, performance evaluation and conclusions.

II. Literature Review

In the packages delivery and pickup field, VRP is an optimization problem. VRP is closely related to the travelling salesman problem (TSP), and the scheduling problem can be simplified to route assignment issues in multiple travellers. The VRP is used to design an optimal route for a set of vehicles to service a set of customers subject to a set of given constraints. The VRP algorithm can be adapted for warehouse operation management in the physical delivery of goods and services.

As an NP-hard problem, the objective function of VRP is to minimize the overall travel cost of all the vehicles that went on the road. The VRP has been widely studied for about 50 years and many variations of the VRP have been developed to deal with the different logistic tasks. The variations are formulated based on types of delivery, the different capacities of transported goods (AMZL and AMXL), the quality of service required and the characteristics of the customers' and drivers' affinity and preferences. One variation of VRP is capacitated VRP (CVRP). In CVRP, a set of identical vehicles located at the central depot should be optimally routed. Each vehicle should only perform one route and the total weight or volume of packages cannot exceed the capacity limit threshold. One of the most widely studied variations in the vehicle routing problem with time window (VRPTW). In VRPTW, each package should be delivered or picked up within a time window to meet customers' demands and this time window is treated as a hard constraint. The goal function of the VRPTW algorithm is to minimize the total travel cost while meeting all hard constraints. If the time window is a soft constraint, the vehicle will be penalized for the earliness or lateness of delivery. Over the years, several similar and heuristic solutions have been proposed and studied for vehicle routing problems with time windows and capacity optimization. Many heuristic solutions based on integer programming have been proposed to resolve VRPTW. Solomon [2] first proposed mixed-integer programming for VRPTW and introduced a well-known benchmark data set for VRPTW. Matteo et al. [3] considered VRPTW as a scheduling problem and proposed an Integer Linear Programming model. This model considered the possibility of merging two orders to optimize the solution. Duygu et al. [4-5] divided VRPTW into the routing stage and scheduling stage and applied the tabu search algorithm and Integer Linear Programming to these two stages, respectively.

Suresh et al. [6] proposed a generic algorithm-based meta-heuristic, which proved best regarding minimizing the number of vehicles utilized. May et al. [7] improved the GA to solve VRPTW by developing the problem-specific crossover and seven different mutation operators. Agardi et al. [8] compared the performance of different ACO algorithms on VRPTW. Meiling et al. [9] proposed an improved ACO algorithm for VRPTW and could obtain a solution with lower cost and higher service quality than the traditional ACO algorithm. Another kind of heuristic for VRPTW is a hybrid optimization algorithm. Zhang et al. [10] introduced simulated annealing into a GA for "premature" convergence. It was shown that their algorithm was better than the original GA. Chen [11] used a discrete particle swarm optimization algorithm to assign customers on routes and a simulated annealing (SA) algorithm to avoid becoming trapped in the local optimum. Zheng et al. [12] used the ACO algorithm to get the local optimization solution and the GA to find the global optimization solution. Pratiwi [12] proposed a hybrid algorithm between cat swarm optimization and crow search, which could get better performance when the population increased. Shen et al. [13] used an ACO algorithm to optimize the solution first and then performed brainstorm optimization with a local search for further optimization. Dong also designed a new genetic representation that focuses on delivery scheduling and vehicle routing. From this design, a suitable crossover and mutation operator is developed.[14]

This paper was also greatly influenced by the above-mentioned combination optimization techniques research papers. Therefore the experiments conducted in this project include regular standalone bio-inspired algorithms, regular bio-inspired algorithms with clustering algorithms, hybrid bio-inspired algorithms and hybrid bio-inspired algorithms with clustering algorithms. The clustering algorithm chosen in this paper is K Means clustering because of its simplicity, scalability and high adaptivity to various problems. ACO and GA were chosen for bio-inspired algorithms and the hybrid GA+AOC algorithm has also been experimented with in this paper.

III. Problem Formulation and Modeling

As mentioned previously, last-mile delivery scheduling and routing problems can be treated as VRPTW. In VRPTW, the packages should be delivered by multiple vehicles and within a time window to satisfy customers. The objective function of VRPTW is to minimize the total travel distance and maximize the customers' satisfaction. The time window constraint is treated as a soft constraint and the vehicle will be penalized for the earliness or lateness of delivery. Each vehicle has the hard constraints of a capacity limit and maximum work time. Each vehicle must start and end the route at the depot. Each package can only be delivered once and by one vehicle.

The VRPTW can be represented as followed: $G = (V, E)$ denotes a connected graph where $V = V_D \cup V_C$ is the set of nodes and $E = \{(i, j)|i, j \in N, i \neq j\}$ is the set of edges. $V_D = \{0, n + 1\}$ are the starting and ending points of each route, which should be the central depot. $V_C = \{1, 2, 3, ..., n\}$ denote the set of drop-off locations.

$V_C = \{1, 2, 3, ..., n\}$ denote the set of drop-off locations. Each drop-off location $i \in V_C$ has a package with volume q_i, a time window $[e_i, l_i]$. The time window of the depot $[e_0, l_0]$ means the available working time of vehicles. Q is the volume capacity of each vehicle $k \in K$ where K is a set of vehicles. For each edge (i, j) in E, t_{ij} and d_{ij} represent the travel time and distance of this edge. e_{ik} and l_{ik} represent the earliness and lateness at the drop-off location i served by vehicle k. epu and lpu are the respective penalty cost per hour for arriving early or later than the time window. c is the cost paid for one meter in the distance. s_{ik} denotes the time that vehicle k reaches the vertex i. h_i denotes the service time of package i.

Given the definition mentioned above, we can provide the decision variable and mathematical model of our problem. The decision variable is as follows:

$$x_{ijk} = \begin{cases} 1 & \text{if } edge(i,j) \text{ is visited by the vehicle } k \\ 0 & \text{otherwise} \end{cases}$$

The mathematical model of our problem is as follows:

$$\min c \sum_{k\in K}\sum_{i\in V}\sum_{j\in V} d_{ij}x_{ijk} + epu \sum_{k\in K}\sum_{i\in V_C} e_{ik} + lpu \sum_{k\in K}\sum_{i\in V_C} l_{ik} \tag{1}$$

Subject to:

$$\sum_{j\in V}\sum_{k\in K} x_{ijk} = 1 \quad i \in V_C \tag{2}$$

$$\sum_{i\in V} x_{imk} - \sum_{j\in V} x_{mjk} = 0 \quad m \in V_C \quad k \in K \tag{3}$$

$$\sum_{i\in V_C} q_i \sum_{j\in N} x_{ijk} \le Q \quad k \in K \tag{4}$$

$$\sum_{j\in V_C} x_{0jk} = 1 \quad k \in K \tag{5}$$

$$\sum_{i\in V_C} x_{i(n+1)k} = 1 \quad k \in K \tag{6}$$

$$s_{ik} + h_i + t_{ij} = s_{jk} \quad i,j \in V \quad k \in K \tag{7}$$

$$e_0 \le s_{0,k} \le s_{n+1,k} \le l_0 \quad k \in K \tag{8}$$

$$e_{ik} = \max(e_i - s_{ik}, 0) \quad i \in V_C \quad k \in K \tag{9}$$

$$l_{ik} = \max(s_{ik} - l_i, 0) \quad i \in V_C \quad k \in K \tag{10}$$

The objective function (1) is to minimize the travelling distance of routes, the earliness penalty and the lateness penalty. Constraints (2) and (3) ensure that each drop-off location is visited once by exactly one vehicle. Constraint (4) guarantees that the total volume of packages in one vehicle doesn't exceed the capacity limit. Constraints (5) and (6) make sure that each vehicle starts and ends at the central depot. Constraint (7) is the relationship between the visit time of a drop-off location and its successor. Constraint (8) guarantees that the vehicle's maximum work time is not exceeded. Constraints (9) and (10) show how to compute the earliness and lateness of each package and ensure that the earliness and lateness are non-negative.

IV. Proposed Solution

This section first describes the Encoding and Decoding Procedure that can transform the VRPTW into a TSP problem. Next, K-means clustering that can pre-assign packages into routes while satisfying the hard constraints is described. Then it describes three different meta-heuristic search algorithms that are applied to this problem. Finally, a rescheduled solution for VRPTW is described.

A. *The Encoding and Decoding Procedure*

Meta-heuristic algorithms are difficult to be applied to VRPTW directly because VRPTW has multiple vehicles, constraints of time window and capacity. Therefore, the VRPTW is transformed into a TSP problem by using the encoding and decoding procedure. The encoding and decoding procedures are inspired by literature [9,16,17]. The meta-heuristic algorithms will search for solutions of TSP form. When computing the objective score, the solution will be transformed into VRPTW form. The encoding procedure transforms the customers of all sub-routes into a form of TSP problem, which is a permutation of all the customers. For example, the first sub-route R1=0, 5, 3, 2, 0, the second sub-route R2=0, 7, 1, 6, 0 and the third sub-route R3=0, 8, 4, 0 are encoded as P=5, 3, 2, 7, 1, 6, 8, 4. The decoding procedure transforms the permutation of all customers into several feasible sub-routes. The decoding procedure considers the hard constraints of capacity and maximum work time of vehicles. The procedure assigns customers to the current sub-route only if equations (4) and (7) in section 2 can be met. Once the constraints cannot be met, the procedure will start a new sub-route and assign customers to the new route.

B. *K-means Clustering*

K-means clustering [15] is an unsupervised classification algorithm. It randomly selects cluster centroids around the region and iteratively clusters the target points. It is utilized to dispatch packages into routes before applying metaheuristics to optimize scheduling. Algorithm 1 shows how it finds the package assignment in the greedy approach.

Algorithm 1 K-means Clustering

```
1: find $\min(n_{clusters})$, $n_{current} = n_{clusters}$
2: while $iteration < max_{iteration}$ do
3:     generate 100 $candidates$ from $Kmeans(n_{current})$
4:     for each $candidate$ in $candidates$ do
5:         merge $candidate$ to $n_{clusters}$
6:         if $candidate$ satisfy constraint then
7:             break
8:         end if
9:     end for
10:    $iteration++$, $n_{current}++$
11: end while
```

Since the drop-off locations of the packages are randomly distributed, the optimal solution is difficult to find. The adopted algorithm clusters them while minimizing the number of routes and maintaining low timing overhead. It calculates the minimum number of routes first as a target to satisfy the hard constraints and minimize cost, then iteratively generates a gradually increasing number of clusters, and attempts to merge them into the target number of clusters while satisfying the constraints to converge.

C. *Genetic Algorithm*

GA [18] is one of the most widely used bio-inspired optimization algorithms. The algorithm is inspired by the biological evolution process. It uses the concepts of "Natural Selection" and "Genetic Inheritance" proposed by Charles Darwin to evolve a more fit solution. The Partially Mapped Crossover and Insert Mutation are used in our implementation of the GA. Algorithm 2 shows the process of the GA.

Algorithm 2 Genetic algorithm

```
 1: init $populations$
 2: while $iteration < max_{iteration}$ do
 3:     for each $parent1$, $parent2$ in $populations$ do
 4:         crossover to generate one child
 5:         delete the parent with larger $score_{objective}$
 6:     end for
 7:     mutate $populations$
 8:     $iteration++$
 9:     extract best solution from $populations$
10: end while
```

D. Adaptive Genetic Algorithm

The original GA had fixed crossover probability and mutation probability. In each iteration, a random number is generated. The crossover and mutation will happen only if the random number is smaller than the crossover or mutation probability. To improve the performance of GA, we added an adaptive operator. The adaptive operator is inspired by [20]. This operator will increase the probability of the occurrence of crossover and mutation if the Adaptive GA (A-GA) algorithm consistently produces better offspring. It will reduce the probability of the occurrence of crossover and mutation if the offspring is worse than the parent. The details of the adaptive operator are as follows:

$$P_c(t+1) = \begin{cases} P_c(t) - 0.05 & \text{if} \frac{\overline{f_{parent}}}{\overline{f_{children}}} - 1 \leq -0.1 \\ P_c(t) & \text{if} -0.1 < \frac{\overline{f_{parent}}}{\overline{f_{children}}} - 1 < 0.1 \\ P_c(t) + 0.05 & \text{if} \frac{\overline{f_{parent}}}{\overline{f_{children}}} - 1 \geq 0.1 \end{cases} \tag{11}$$

$$P_m(t+1) = \begin{cases} P_m(t) - 0.005 & \text{if} \frac{\overline{f_{parent}}}{\overline{f_{children}}} - 1 \leq -0.1 \\ P_m(t) & \text{if} -0.1 < \frac{\overline{f_{parent}}}{\overline{f_{children}}} - 1 < 0.1 \\ P_m(t) + 0.005 & \text{if} \frac{\overline{f_{parent}}}{\overline{f_{children}}} - 1 \geq 0.1 \end{cases} \tag{12}$$

E. Ant Colony Algorithm

ACO [19] algorithm is a population-based metaheuristic modelled on the actions of an ant colony. The algorithm uses artificial ants and pheromones to simulate real ant colonies and find optimal solutions. In our implementation, a tabu list of visited nodes is maintained for each ant to prevent loops. The pheromone deposits are inversely proportional to the objective score of the solution constructed by the ant. The algorithm considers the global information and time cost by using objective scores of complete paths to update the pheromone. Algorithm 3 shows the process of the ACO.

Algorithm 3 Ant Colony algorithm

```
 1: init $pheromones$ tails
 2: while $iteration < max_{iteration}$ do
 3:     for each $ant$ do
 4:         construct solution
 5:         compute $score_{objective}$
 6:     end for
 7:     update $pheromones$
 8:     $iteration++$
 9:     extract best solution from $ants$
10: end while
```

F. Hybrid Ant Colony Algorithm

The hybrid ACO (H-ACO) used in this paper is proposed by Zheng et al [17]. This hybrid method combines GA with improved ACO to get a better solution. The method first uses improved ACO to gain local optimization in each iteration. Then it makes use of GA to gain global optimization. Compared with the original ACO, the improved ACO adds a pheromone limit and changes the update rule of the pheromone. In simple ACO, the arc of optimal or sub-optimal routes may have excessive growth, which can make all the ants get the same result. To avoid this effect, the pheromone value is set to be in an interval. The pheromone update is the positive feedback mechanism in ACO. To increase the overall ants' searching capabilities and avoid the fast partial optimum, only the best-so-far ant is allowed to update the pheromone after each iteration. The updated rules are shown in (13) and (14):

$$\tau_{ij}(t+1) = \rho\tau_{ij} + \delta\tau_{ij}^{gbest} \quad \rho \in (0,1) \tag{13}$$

$$\tau_{ij}^{gbest} = \begin{cases} \frac{1}{L_{gbest}} & \text{if ant go through } arc(i,j) \text{ in circle} \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

Where τ_{ij} is the pheromone of $arc(i,j)$, ρ is the trail persistence and L_{gbest} is the global best solution.

The algorithm adds a GA operator into the ACO to improve convergence speed and global search capacity. After each iteration of ACO, the crossover and mutation are applied to generate new possible optimal solutions. The Partially Mapped crossover is applied to the global best solution and the best solution of this iteration. The insert mutation is applied to the global best solution at random times. The global best solution will be updated if there is a better solution in the process of crossover and mutation.

G. Reschedule Solution

The rescheduled solution was inspired by an initialization algorithm for VRPTW proposed by Tas et al [4]. Rescheduling happens when a vehicle fails to deliver a package, so packages have already been assigned to vehicles to meet the hard constraint of a capacity limit. Therefore, the rescheduled solution only needs to consider the time window constraint for one vehicle that fails to deliver a package.

Let P denote all the packages in the vehicle that need to be rescheduled. P can be divided into two parts: P_p (packages that have been delivered) and P_f (packages that need to be delivered). The package that needs to be delivered can be inserted between each of the two packages in P_f. The hard constraint of maximum work time will be checked for each possible insertion. Then the new cost of each possible insertion will be computed. The solution that meets the constraints of work time and has the lowest cost will be chosen as the final reschedule solution.

V. Performance Evaluation

The experiment configuration and results are described and discussed in this section.

A. Dataset

To evaluate the optimizers, a dataset compiled and constructed from the 2021 Amazon Last Mile Routing Research Challenge Dataset [21] is implemented, including (i) 158 packages with their dimensions, geographical data of designated drop-off locations, preferred time windows, and expected service time; (2) executive vehicles with their volume, geographical data of starting distribution center, and working time windows; (3) travelling times between each pair of drop-off locations and distribution center to drop-off locations.

B. Experiment Configuration

Eight experiments are conducted in total, including GA, A-GA, ACO, H-ACO and their K-means clustering hybrid versions. The iterative costs from the optimization are recorded for a one-hour duration for the eight experiments. The test-bench code is compiled with Python 3.8 ipykernel and executed on an Intel Core i7-10870H processor with 8 cores and a 2.20 GHz to 5.00 GHz frequency. The Python built-in $concurrent.futures$ module is used to parallelize the optimizer after K-means clustering preprocessing.

After tunning, the parameters for the metaheuristics are configured as follows: For the GA and the A-GA optimizers, the population is set to 100, mutation probability is set to 0.3, and crossover probability is set to 0.7. For the ACO and the H-ACO optimizers, the number of ants is set to 100, α is set to 1, β is set to 5, ρ is set to 0.1, and Q are set to 800.

C. Cost Reduction Capability

The initial cost is 900585, which is calculated from the initial order of the packages. The last reported costs from the optimizers at the end of the one-hour experiments are recorded, as the following table shows. Each test is conducted for 5 times and the average values are calculated.

TABLE I
The final cost obtained from the experiments (Typical)

	GA	A-GA	ACO	H-ACO
Baseline	446237	452023	415202	402457
Hybrid	385239	371423	325150	342110

The baseline ACO hybrid with the K-mean Clustering achieves the best cost reduction, which is 563415. However, the H-ACO does not benefit from the crossover and mutation to get much performance improvement but is affected by more computation overhead. Compared with the fixed-parameter GA, A-GA has a similar capability but it is still relatively inefficient for this problem compared to ACO and very unstable.

The K-means Clustering helps to further improve the cost reduction capability of all the metaheuristics. Each hybrid version of the optimizers achieves a better final cost. When the algorithm preprocesses the packages, it attempts to ensure that the locations with a short distance from each other are in the same cluster, thus, it can not only reduce the problem size but also helps the optimizers to reduce the travelling costs.

D. Convergence Speed

The local minimum costs are recorded iteratively while conducting the experiments, one typical result obtained from the experiments is selected for plotting the trends, as figure 1 shows.

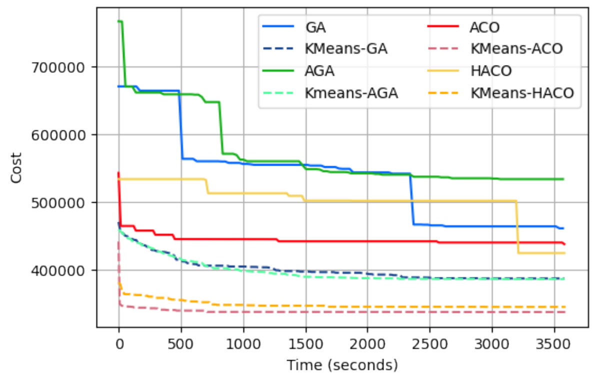

Fig. 1. The overall cost versus time graph (Typical).

In general, ACO and H-ACO achieves the fastest convergence speed and are more robust. The high randomness of GA and A-GA cause their performance relatively unstable and to trapped in local minima for many iterations when handling large dataset. The A-GA does not perform its advantages in the experiments, which may cause by its insufficient cost reduction capability in such problems.

Moreover, benefit from the smaller problem size and parallelization, the K-means Clustering version of the optimizer has the similar convergence speed to achieve a much better cost reduction.

E. Preprocessing Study

As the figure 2 shows, there are more closer drop-off locations on each route, which is significantly helpful on reducing the cost and convergence time.

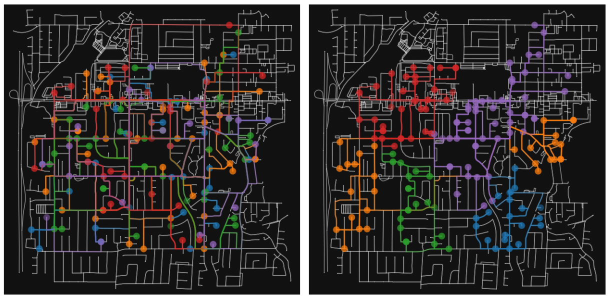
Fig. 2. The routing result after GA optimization. (A) Baseline (B) Hybrid

VI. Conclusions and Recommendations

We have proposed two hybrid optimization metaheuristics with their adaptive and modified versions for last-mile delivery scheduling problems and evaluated the algorithms on a real dataset. The hybrid metaheuristics adopt a K-means clustering algorithm to pre-assign the packages to the delivery vehicles before scheduling. An algorithm to reschedule packages that failed on the first attempt is integrated as well. Overall, the ACO optimizer has advantages over GA in both speed and performance respects. The adopted K-means Clustering algorithm demonstrated its value in improving the convergence speed and cost reduction capability of the optimizer, which introduces high scalability to the hybrid metaheuristics. It provides a prerequisite for the application of massive parallelization for larger datasets to bring huge performance improvements.

Currently speaking, we recommend utilizing the ACO algorithm hybrid with K-means clustering to optimize the problem. The future work of this project includes:
(i) More precisely tuning the parameters of the algorithm and making comparisons to decide better configurations.
(ii) Adopting advanced classification algorithms instead of K-means Clustering to pre-assign packages into vehicles.
(iii) Exploits the parallelism of the optimizers by introducing larger datasets.

References

[1] J. Gaubys, "Ecommerce share of retail sales (2021–2026)". Oberlo. https://www.oberlo.com/statistics/ecommerce-share-of-retail-sales (accessed Dec. 16, 2022).

[2] M. M. Solomon, "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints," *Operations Research*, vol. 35, No. 2, pp. 166-324, April 1987

[3] M. Cosmi, G. Nicosia, and A. Pacifici, "Scheduling for Last-mile Meal-delivery Processes," *IFAC-PapersOnLine*, vol. 52, issue. 13, pp. 511-516, August 2019

[4] D. Tas, N. Dellaert, T. V. Woensel, and T. D. Kok, "Vehicle Routing Problem with Stochastic Travel Times Including Soft Time Windows and Service Costs," *Computers & Operations Research*, vol. 40, issue. 1, pp. 214-224, January 2013

[5] D. Tas, O. Jabali, and T. V. Woensel, "A Vehicle Routing Problem with Flexible Time Windows," *Computers & Operations Research*, vol. 52, Part. A, pp. 39-54, December 2014

[6] S. N. Kumar and R. Panneerselvam, "A Comparative Study of Proposed Genetic Algorithm-Based Solution with Other Algorithms for Time-Dependent Vehicle Routing Problem with Time Windows for E-Commerce Supply Chain," *Journal of Service Science and Management*, vol. 8, pp. 844-859, 2015

[7] A.T. May, C. Jariyavajee, and J. Polvichai, "An Improved Genetic Algorithm for Vehicle Routing Problem with Hard Time Windows," *ICECET*'21, December 2021

[8] A. Agardi, L. Kovacs, and T. Banyai, "Ant Colony Algorithms For The Vehicle Routing Problem With Time Window, Period And Multiple Depots," *Manufacturing Technology*, vol. 21, issue. 4, pp. 422-433, 2021

[9] M. He, Z. Wei, X. Wu, and Y. Peng, "An Adaptive Variable Neighborhood Search Ant Colony Algorithm for Vehicle Routing Problem With Soft Time Windows," *IEEE Access*, vol. 9, pp. 21258-21266, February 2021

[10] X. Liu, Z. Jiang, and J. Han, "Hybrid Genetic Simulated Annealing Algorithm with its Application in Vehicle Routing Problem with Time Windows," *Advanced Materials Research*, vol. 148-149, pp. 395-398, October 2010

[11] W. Chen, P. Dai, Y. Chen, D. Chen, Z. Jiang, "Logistics Distribution Vehicle Routing Problem with Time Windows," *Advanced Materials Research*, vol. 468-471, pp. 2047-2051, February 2012

[12] A. B. Pratiwi, "A hybrid cat swarm optimization - crow search algorithm for vehicle routing problem with time windows," *ICITISEE*'17, November 2017

[13] Y. Shen, M. Liu, J. Yang, Y. Shi, and M. Middendorf, "A Hybrid Swarm Intelligence Algorithm for Vehicle Routing Problem With Time Windows," *IEEE Access*, vol. 8, pp. 93882-93893, June 2020

[14] M. L. Pilat and T. White, "Using Genetic Algorithms to Optimize ACS-TSP," *Ant Algorithms*, p282-287, urlhttps://doi.org/10.1007/3-540-45724-0_28

[15] C. Piech, "K Means". Stanford CS221. https://stanford.edu/~cpiech/cs221/handouts/kmeans.html (accessed Dec. 16, 2022).

[16] J. Jia, Y. Chen, Y. Liu and A. Khamis, "Trip Itinerary Planning: A Bio-inspired Metaheuristic Approach," *2022 IEEE International Conference on Smart Mobility (SM)*, 2022, pp. 32-37, doi: 10.1109/SM55505.2022.9758204.

[17] L. J. Zheng, D. C. Dong and D. Y. Wang, "A hybrid intelligent algorithm for the vehicle scheduling problems with time windows," *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, pp. 2756-2761, doi: 10.1109/ITSC.2014.6958131.

[18] J. H. Holland, "Genetic Algorithms and Adaptation," *Adaptive Control of Ill-Defined Systems, NATO Conference Series*, vol 16, 1984, urlhttps://doi.org/10.1007/978-1-4684-8941-5_21

[19] C. Alberto, D. Marco, and M. Vittorio, "Distributed Optimization by Ant Colonies," *Proceedings of the First European Conference on Artificial Life*, 1991.

[20] D. W. Cho, Y. H. Lee, T. Y. Lee, and M. Gen, "An adaptive genetic algorithm for the time dependent inventory routing problem", *Journal of Intelligent Manufacturing*, vol. 25, pp. 1025-1042, 2014, https://doi.org/10.1007/s10845-012-0727-5

[21] D. Merchan, J. Arora, J. Pachon, K. Konduri, M. Winkenbach, S. Parks, and J. Noszek, "2021 Amazon Last Mile Routing Research Challenge: Data Set", *Transportation Science*, September 2022, url-https://doi.org/10.1287/trsc.2022.1173.