A Model for the Capacitated Vehicle Routing Problem with Pickup and Delivery Considering Package Returns

Patarapon Jumpahom, Wachirawit Chaphumee, Siwanart Tanongjit, and Monsikarn Jansang*
Mahidol Wittayanusorn School Nakhon Pathom, Thailand
**pattarapon.jam_g33@mwit.ac.th*

Abstract

Following the COVID-19 pandemic, online purchasing has grown rapidly. The logistics industry has prioritized route planning, which relates to the Capacitated Vehicle Routing Problem with Pickup and Delivery (CVRPPD), to reduce operational costs. However, existing studies have not yet examined route planning in cases where packages are returned due to reasons such as incorrect addresses, absence of the recipient, or refusal of acceptance. This research, therefore, develops a mathematical model for logistics route planning with capacity constraints, which supports both pickup and delivery while accounting for the probability of package returns. The authors studied maps of various sized cities in the United States and found that the probability of a package return is correlated with the probability of delivery failure followed a sigmoid curve. The calculated probability of failure showed similar results across all routing methods. In terms of distance and processing time, the Saving Algorithm yielded results comparable to the Nearest Neighbor Heuristic but significantly lower than Simulated Annealing. Regarding the probability of failure, the Saving Algorithm provided slightly lower results than the other methods. This mathematical model can be practically applied to route planning, helping to increase efficiency and reduce transportation costs, while also providing assurance to operators regarding package transportation in the event of returns.

Keywords: Capacitated Vehicle Routing Problem with Pickup and Delivery (CVRPPD); Simulated Annealing; Saving Algorithm; Nearest Neighbor Heuristic; Mathematical model; Returned package

I. INTRODUCTION

The Vehicle Routing Problem (VRP) has been a subject of study for several decades (Dantzig & Ramser, 1959). It focuses on finding the optimal route for vehicle scheduling. However, variations in the definition of an "optimal route," along with increasing auxiliary factors, have led to a variety of approaches for finding suitable routes, since in the real world, transportation often encounters various issues such as traffic conditions, accidents, and road damage.

Here we focus on the problem of package returns. These are defined as failed deliveries, where the package either remains in the vehicle and must be brought back to the distribution center, or enters the return process later. This problem arises due to incorrect addresses, the absence of the recipient, or the recipient's refusal for various reasons. Package retention in a capacity-constrained vehicle can lead to errors in the delivery schedule for the Capacitated Vehicle Routing Problem for Pickup and Delivery (CVRPPD) during transport. This research aims to study the route distance and test the feasibility of rerouting from the original plan *before* the start of the trip, as failed deliviers can prevent pickups along the route.

Package returns from online orders are a common problem in many countries. A report by Loqate (2021) found that failed deliveries from online purchases accounted for 8%, 6%, and 7% in the United States, the United Kingdom, and Germany, respectively, in 2020. The total annual cost calculated from returned packages in these three countries was approximately 16 million Thai Baht.[1] This demonstrates that package returns impact both cost and transportation route management, posing a significant current challenge for the global logistics industry.

Currently, several methods are used to construct routes for solving vehicle routing problems. This study discusses three popular methods: Nearest Neighbor Heuristic (NNH), Savings Algorithm, and Simulated Annealing (SA). Their general procedures and characteristics are:

Nearest Neighbor Heuristic (NNH) is a simple method that selects the next closest point from the current location to build a route, repeating the process until all points are covered. The advantage of this method is its low complexity and fast processing. However, research by Harahap (2023) indicates that NNH often yields longer distances compared to other methods.

Savings Algorithm is a process used to construct vehicle routes by merging two routes to create a new one, based on comparing the distance to the original routes. If the new merged route results in a shorter total distance, that new route is selected. This method typically results in routes with shorter distances while simultaneously reducing processing time (Paessens, 1988).

Simulated Annealing (SA) improves existing routes through a stochastic process that considers a Temperature parameter. In the initial phase of the process, when the temperature is high, the method still has a chance to select and continue with a new route, even if it slightly increases the total distance. However, as the temperature decreases after multiple iterations, the method only accepts routes that result in a shorter total distance. Wei (2018) found that Simulated Annealing yields better results than Tabu Search and Genetic Algorithm when used to solve the Capacitated Vehicle Routing Problem (CVRP).

These three methods possess distinct characteristics and limitations. A comparison of these three methods will help to identify which method is best suited for improving vehicle routes in different scenarios.

II. METHODS

1: Defining the CVRPPDRP Model

To address the stated problem, it is necessary to understand its characteristics. The authors adapted the mathematical model of the Capacitated Vehicle Routing Problem with Pickup and Delivery (CVRPPD), which involves the dual tasks of package delivery and pickup, originally studied by Kui-Ting Chen et al. (2015). This model was refined by incorporating additional variables and constraints to account for package returns, resulting in a mathematical model that aligns with the research objectives: the Capacitated Vehicle Routing Problem with Pickup and Delivery for Returned Packages (CVRPPDRP).

Graph: A graph is defined as $G = (V, E)$, where:

- $V = \{0,1,2,\dots,n\}$: The set of all vertices (nodes), representing road intersections which are proxies for customer locations. Node 0 is defined as the distribution depot.
- $E = \left\{ \{i,j\} \middle| \begin{matrix} i,j \in V, i\sim j \\ if\ there\ is\ a\ road\ connecting\ i\ and\ j \end{matrix} \right\}$: The set of edges, representing roads connecting intersections.

Sets

- $P \subseteq V - \{0\}$: The set of pickup nodes.
- $D \subseteq V - \{0\}$: The set of delivery nodes.
- $P \cap D = \emptyset$: A condition that states the sets of pickup and delivery nodes have no common members.
- $SB \subseteq D$: The set of delivery nodes that may result in returned packages.

[1] Loqate (2021) states that this cost occurred in the U.S. ($193,730), the U.K. (£68,084), and Germany (€144,354). When converted using 2020 exchange rates, the total is approximately 16 million Thai Baht per year.

Parameters

- c_{ij}: The travel distance from node i to node j.
- q_i: The change in the number of packages in the vehicle at node i (negative for delivery, positive for pickup).
- Q: The maximum package capacity of the delivery vehicle.
- p: The probability of a package return occurring at delivery node .

Decision Variables

- x_{ij}: A Binary variable equal to 1 if the vehicle travels along the path from node i to node j.
- u_i: The load (number of packages) in the vehicle after passing node i.
- f_i: A binary variable equal to 1 if a failure occurs at node because the vehicle cannot pick up a package due to reaching full capacity, and 0 otherwise.
- r_i: A binary variable equal to 1 if node i is a return node ($i \in SB$)

Objective Function

$$Minimize \quad \sum_{i\in V}\sum_{j\in V}(c_{ij} \cdot x_{ij}) \qquad (1)$$

$$Minimize \sum_{i\in SB}(p \cdot f_i) \qquad (2)$$

Equation (1) minimizes total travel distance, while equation (2) quantifies delivery-failure probability due to returns $\boldsymbol{p}$.

Constraints

$$\sum_{j\in V} x_{ij} = 1 \quad \forall i \in P \qquad (3)$$

$$\sum_{j\in V} x_{ij} = 1 \quad \forall i \in D \qquad (4)$$

$$\sum_{j\in V} x_{0j} = 1 \qquad (5)$$

$$\sum_{i\in V} x_{i0} = 1 \qquad (6)$$

$$\sum_{j\in V} x_{ij} = \sum_{j\in V} x_{ji} \quad \forall i \in V \qquad (7)$$

$$0 \le u_i \le Q \quad \forall i \in V \qquad (8)$$

$$u_i + q_j \le u_j + Q \cdot (1 - x_{ij}) \quad \forall i, j \in V \qquad (9)$$

$$u_j \ge u_i + q_j(x_{ij} - r_j) - Q \cdot (1 - x_{ij}) \quad \forall i, j \in V \qquad (10)$$

$$p = P(i \in SB), \quad \forall i \in D \qquad (11)$$

$$Q \ge u_i \cdot (1 - f_i), \quad \forall i \in P \qquad (12)$$

- Equations (3) and (4) specify that the vehicle visits each pickup and delivery node exactly once.
- Equations (5) and (6) state that the vehicle must begin and end its journey at the distribution depot (node 0).
- Equation (7) ensure path continuity (if a vehicle arrives at a node, it must depart from that node).
- Equation (8) limits the load inside the vehicle to no less than zero and no more than the maximum capacity.
- Equation (9) prevents subtours (routes that do not include any customers or do not start and end at the depot) by employing Miller-Tucker-Zemlin constraints (Desrochers, M. & Laporte, G., 1991).
- Equation (10) controls the package load in the vehicle both during route construction and for verification after simulating return events.
- Equation (11) defines the probability p of a package return at a delivery node (D).
- Equation (12) ensures that the load at a pickup node i (in set P) does not exceed the capacity, Q. If it does, a failure ($f_i = 1$) is registered.

After the route is constructed, Equations (2), (11), and (12) are applied to calculate the probability of failure. The primary focus of this research is to find the shortest route, with the secondary benefit being the minimization of the probability of transport failure.

2: Defining City Data and Study Points

Data from the National Household Travel Survey, USA, indicates that the average household typically orders packages online once a week (Cokyasar, T., 2022). In this study, road intersections are used as proxies for customer locations. In this study, each road intersection is treated as a representative household location. For analyzing deliveries on a single day, one-seventh of all intersections are randomly selected as pickup or delivery points.

City map data are imported from OpenStreetMap and converted into an edge-weighted graph for which nodes represent intersections, edges represent road segments, and edge weights represent the road distance between intersections. One node is randomly selected as the depot. The remaining nodes, totaling one-seventh of the total number of nodes, are designated as customer nodes. One-fifth of the customer nodes are designated as pickup nodes (P), and the remaining nodes are designated as delivery nodes (D). A sample map is shown in Figure 1.

The demand at each node, (change in package load), is defined: delivery nodes () are assigned a value of , and pickup nodes are assigned a random value between 1 and 4. Next, the demand at each node is defined—i.e., the load change q_i at point i. Delivery nodes (D) are assigned a demand of −1, while pickup nodes have randomly assigned values from 1 to 4. Vehicle capacity (Q) is set to the larger of the total number of parcels to be delivered and the total number of parcels to be picked up. This reflects practical daily operations: the logistics system knows each morning how many parcels must be delivered and how many will be picked up, and selecting a vehicle with appropriate capacity increases delivery efficiency. The probability of package return is also defined to align with real-world scenarios where incorrect delivery or other issues can lead to package returns.

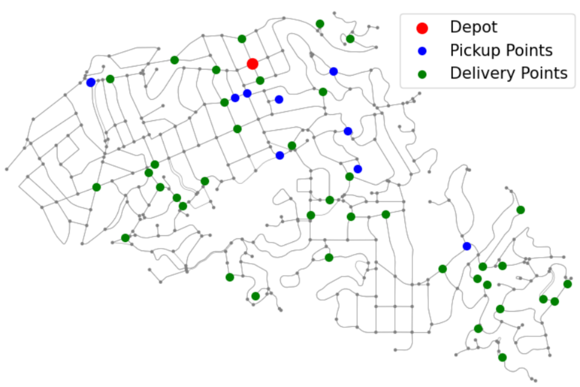

Figure 1. A sample map of Piedmont, California, USA, created with OSMnx.

3: Route Construction Methods

All routes are defined as starting and ending at the depot, and the following route-finding methods were selected for study:

3.1 **Nearest Neighbor Heuristic (NNH)** Route construction using NNH starts at the depot and proceeds to the closest neighboring node from the current location. This is repeated until the last node in the set of all nodes (V) is reached. To satisfy the model's requirement for a closed path, the final node is defined as the depot.

3.2 **Saving Algorithm** Following Clarke, G. and Wright, J.R. (1964) and Namfon Papun and Phattranit Kaewpradit (2019), the Saving Algorithm proceeds as follows:

1. **Computing savings values:** A distance matrix (c_{ij}) between all nodes is first created using nx.shortest_path_length, a function in NetworkX that employs the Dijkstra algorithm to find the shortest distance. Initially, two separate routes are formed: from the depot (0) to customer i and back ($0, i, 0$), and from the depot to customer j and back ($0, j, 0$). The savings value for connecting customers i and j is computed as the reduction in cost achieved by combining the two routes into a single route ($0, i, j, 0$), as shown in Figure 2 (b), compared to the cost of two separate routes ($0, i, 0$) and ($0, j, 0$), as shown in Figure 2 (a). The savings value is calculated using Equation (13):

$$S_{ij} = d_{0i} + d_{0j} - d_{ij} \tag{13}$$

where 0 represents the depot, S_{ij} is the saving between customers i and j, and d_{ij} is the distance between customers i and j.

2. **Sorting the saving values:** The values of are sorted from highest to lowest.

3. **Inserting new customers into existing routes:** If the saving value is positive, meaning the combined route's total distance is shorter than the sum of the two original routes, the two routes are merged into a new single route. Customers i and j are combined into the same transportation route $(0, i, j, 0)$, as shown in Figure 2 (b).

4. **Repeat:** The process of inserting customers is repeated until only one continuous route remains.

3.3 **Simulated Annealing (SA)** Simulated Annealing (SA) is used to find the best route from the depot through all the pickup and delivery nodes. The process begins with generating a random initial route. The route is then repeatedly improved by performing two-opt swaps (swapping the positions of two nodes) and calculating the new distance. If the new route is shorter, it is accepted. If the distance is longer, the route is accepted with a probability that decreases according to the cooling rate. The cooling process reduces the chance of accepting worse routes, which helps the algorithm avoid local minima and find the global best solution. For this study, the initial 'temperature' was set to 10,000, the cooling rate was 0.995, and the minimum temperature was 10^{-3}.

(a) (b)

Figure 2. (a) Two separate round trips for delivery to all points. (b) Combining delivery points into one route.

4: Route Analysis

Algorithm 1: Failure Simulation

```
1. Define FUNCTION 'simulate_failures(route, q_i, return_prob, vehicle_capacity, seed_value)':
  2. Set random seed to seed_value
  3. Initialize load = number of delivery nodes
  4. Initialize failures = 0
  5. For each node in route:
    6. IF node is in pickup_nodes THEN:
      7. Retrieve demand from q_i[node]
      8. Increment load by demand
    9. ELSE IF node is in delivery_nodes THEN:
      10. Retrieve delivery_amount from q_i[node]
      11. Incremnt load by delivery_amount
      12. IF random value < return_prob THEN:
        13. Increment load by 1 (simulate product return)
    14. IF load > vehicle_capacity OR load < 0 THEN:
      15. Increment failures by 1
      16. BREAK the loop
  17. RETURN failures
18. Initialize total_failures = 0
19. Initialize total_tests = 0
20. For seed_value = 1 to 10001:
  21. Initialize failures_per_seed = 0
    22. Call 'simulate_failures(route_full, q_i, return_prob, vehicle_capacity, seed_value)'
    23. Increment failures_per_seed by returned failures
  24. Increment total_failures by failures_per_seed
  25. Increment total_tests by num_tests_per_seed
26. Calculate per_failure_rate = (total_failures / total_tests) * 100
27. Output per_failure_rate
```

Description of key variables

- route: Sequence of nodes in the route.
- q_i: Demand at node i.
- return_prob: Probability of a package return.
- vehicle_capacity (Q): Maximum vehicle capacity ().
- seed_value: Random seed used for repeated data generation.
- total_failures: Total count of failures across all random seeds.
- total_tests: Total number of tests performed.
- per_failure_rate: Percentage of delivery failures.
- pickup_nodes: Pickup nodes (P).
- delivery_nodes: Delivery nodes (D).
- route_full: The complete route used in the test.

The data obtained from each routing method is analyzed by testing each route against scenarios involving package returns at various points. The probability of package return is set to 0.3, with all delivery nodes having the same return probability. (The value 0.3 is a selected parameter for this study; a realistic value should be used for practical application.)

A transport is considered an immediate failure if a return event prevents the vehicle from being able to perform a subsequent pickup according to the original route plan. After the tests, the resulting values—total distance and the percentage probability of transport failure—are compared across the two objective functions. This analysis is based on Algorithm 1, which is derived from Equations (11) and (12) and implemented as the simulated_failures function. The function begins by initializing the vehicle load (load) to the sum of all packages to be delivered and setting the failure counter (failures) to 0.

For every node in the sequential route, if the node is a pickup node, the vehicle load is increased by the pickup demand, q_i. If the node is a delivery node, a random real number between 0 and 1 is generated. If the random value is less than the return probability, a return occurs, and the package load in the vehicle remains unchanged, but if a return does not occur, the package load is decreased by 1. After each transaction, the vehicle load is checked. If the load does not exceed the vehicle capacity, the process loops to the next node in the route. If the load exceeds the capacity or drops below 0, the loop immediately terminates, and the failure value is recorded as 1. If the route is completed successfully, the failure value is recorded as 0. Each route is tested 10,000 times using different random seeds. The total failures are summed and divided by the total number of tests to yield the percentage probability of failure due to package returns.

III. RESULTS AND DISCUSSION

1: Sample City Data

The OSMnx library was used to import city data from OpenStreetMap to study the optimal route-finding model for package delivery, considering the case of package returns. The size of the sample city was used as a variable to compare the model's performance, which may differ across various city sizes defined by the number of road intersections (nodes). From Table 1, a total of 23 cities in the United States were studied, with city sizes ranging from 6.80 to 29.64 square kilometers.

Name of Sample City	Total Number of Inter-sections	Number of Customer Nodes	Number of Edges (Roads)
Crested Butte, CO	98	14	292
Leavenworth, WA	113	16	319
Calistoga, CA	221	31	558
Moab, UT	239	34	620
Sausalito, CA	258	36	635
Marfa, TX	272	38	888
Carmel-by-the-Sea, CA	283	40	881
Jackson, WY	334	47	916
Whitefish, MT	319	45	867
Piedmont, CA	352	50	944
Breckenridge, CO	373	53	875
Healdsburg, CA	455	65	1,146
Bar Harbor, ME	603	86	1,430
Park City, UT	606	86	1463
Taos, NM	620	88	1,500
Boerne, TX	718	102	1736
Ouray, CO	749	107	1775
Big Bear Lake, CA	866	123	2220
Ashland, OR	927	132	2497
Boulder City, NV	976	139	2358
Truckee, CA	1027	146	2420
Sedona, AZ	1032	147	2385
Los Gatos, CA	1082	154	2441

Table 1. Data on the cities used for the study.

2: Repetitive Testing for Mean Value

After analyzing the data and obtaining three test results for each city, repetitive testing was performed to prevent experimental bias, as the routes were generated from different random samplings. The mean values were recorded for subsequent steps.

To determine the required number of repetitions for minimizing the error in the mean value, the city of Crested Butte, Colorado (with a total of 98 inter-sections) was selected for this preliminary test. The number of repetitions was increased from 1 to 100, and the mean and standard deviation were calculated at each step.

The results from 100 repetitions showed that the outcomes for all the graphs in Figures 3-8 (mean total distance, mean probability of failure, mean processing time, and standard deviation for each variable) tended towards a steady-state condition as the number of repetitions increased.

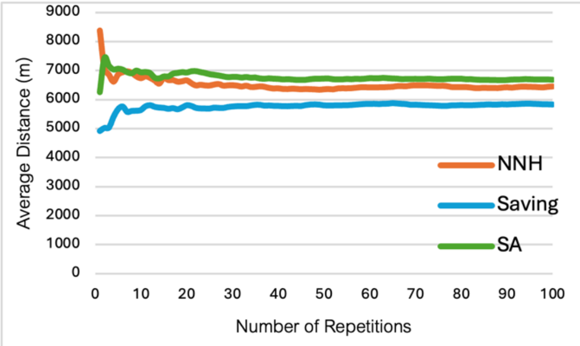

Figure 3. Number of repetitions and average total route distance for the three methods tested for Crested Butte.

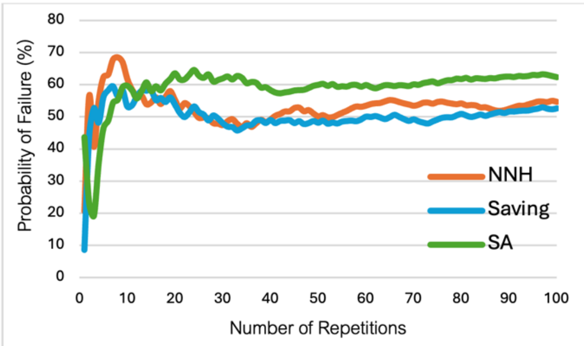

Figure 4. Number of repetitions and average probability of failure for Crested Butte.

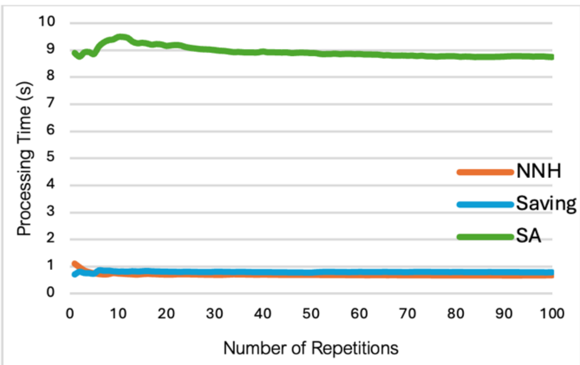

Figure 5. Number of repetitions and average processing time for Crested Butte.

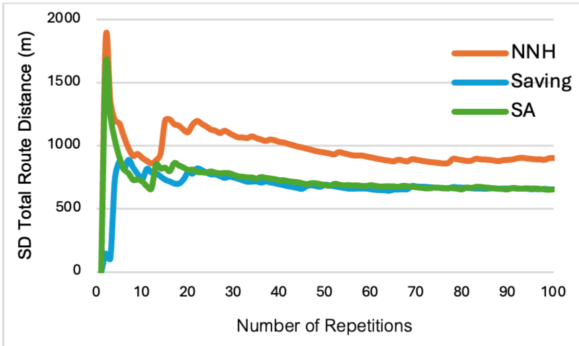

Figure 6. Number of repetitions and standard deviation of the total route distance for Crested Butte.

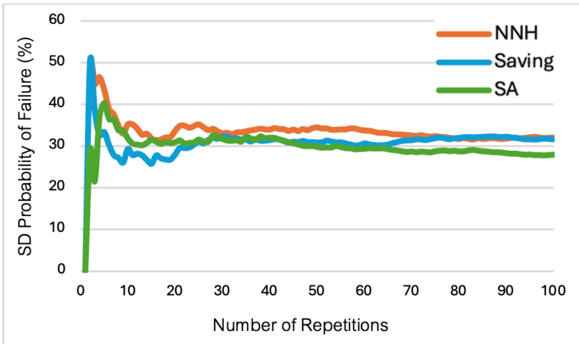

Figure 7. Number of repetitions and standard deviation of the probability of failure for Crested Butte.

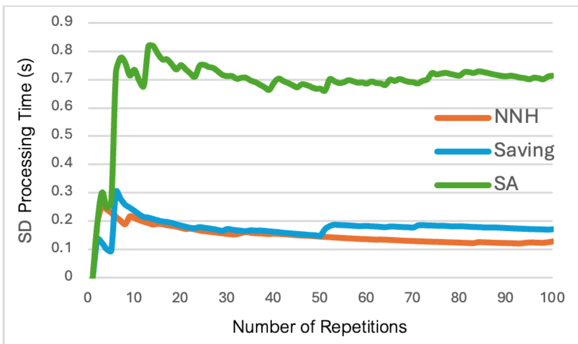

Figure 8. Number of repetitions and standard deviation of the processing time for Crested Butte, Colorado.

Initially, especially within the first 10–20 repetitions, the values for algorithms SA and NNH exhibited higher fluctuations than the Saving Algorithm for some variables. This reflects a higher initial sensitivity to change and instability of the variables early on. However, after approximately 50 repetitions, the results for all algorithms and variables clearly showed a reduction in variance and entered a high-stability range.

The standard deviation graphs (Figures 6-8) demonstrated that all three algorithms tended to reduce the volatility of their results with more repetitions. The Saving Algorithm exhibited the lowest variance from the beginning, while SA and NNH, despite starting with higher volatility, achieved stability later. This suggests that at approximately 50 repetitions, all three models provide stable and reliable values. Therefore, increasing the number of repetitions beyond 50 is unlikely to yield significant changes in the results, making 50 an appropriate threshold to reduce testing time and resources without compromising data accuracy.

Name of Sample City	Number of Inter-sections	Total Route Distance (m)		
		NNH	Saving	SA
Crested Butte, CO	98	6594.26	5975.48	5857.30
Leavenworth, WA	113	7934.05	7073.88	6640.79
Calistoga, CA	221	20340.68	18356.76	18305.49
Moab, UT	239	25300.93	22011.64	21580.02
Sausalito, CA	258	21316.74	18438.08	19143.54
Marfa, TX	272	15566.37	13955.38	13552.38
Carmel-by-the-Sea, CA	283	14647.64	12624.75	12563.87
Jackson, WY	334	33922.35	28705.02	29893.32
Whitefish, MT	319	32454.29	28564.28	28873.43
Piedmont, CA	352	25705.40	21772.30	23311.06
Breckenridge, CO	373	66256.69	59060.36	67362.94
Healdsburg, CA	455	41012.65	35388.94	43241.81
Bar Harbor, ME	603	127145.38	109821.84	149619.99
Park City, UT	606	88204.72	77903.88	105835.20
Taos, NM	620	63072.84	52873.42	73279.52
Boerne, TX	718	84543.07	73762.75	129359.45
Ouray, CO	749	409919.77	344498.56	512178.44
Big Bear Lake, CA	866	82151.38	68491.63	123,928.8 9
Ashland, OR	927	71781.17	62782.27	122488.43
Boulder City, NV	976	101276.22	85590.53	171532.38
Truckee, CA	1027	181676.95	96398.51	398599.84
Sedona, AZ	1032	111506.39	162393.00	192711.14
Los Gatos, CA	1082	118229.50	101999.99	208874.46

Table 2. The route distance for each city generated by the three different routing methods.

3: Experimental Results

The data on the total number of intersections (nodes) is treated as representing all house locations on the roads between those intersections, simplifying the analysis. City data was then defined, and routes were generated.

Table 2 presents the route distances for each method. It was found that the distance generated by each method is directly proportional to the total number of intersections. Two cities—Ouray, CO and Truckee, CA—were found to be outliers with unusually high distances.

4: Analysis of Experimental Results

The charts and tables presented below clearly demonstrate that each route generation method

Name of Sample City	Number of Inter-sections	Probability of Failure (%)		
		NNH	Saving	SA
Crested Butte, CO	98	51.01	52.23	59.27
Leavenworth, WA	113	70.73	57.36	63.68
Calistoga, CA	221	74.07	64.55	71.11
Moab, UT	239	60.27	67.88	63.26
Sausalito, CA	258	67.47	69.66	62.93
Marfa, TX	272	66.66	58.40	63.44
Carmel-by-the-Sea	283	63.15	58.89	66.19
Jackson, WY	334	71.26	74.44	71.75
Whitefish, MT	319	61.10	54.70	69.30
Piedmont, CA	352	65.11	62.33	75.44
Breckenridge, CO	373	70.64	60.27	69.32
Healdsburg, CA	455	74.29	78.79	67.26
Bar Harbor, ME	603	74.41	54.25	65.60
Park City, UT	606	73.48	63.07	69.60
Taos, NM	620	69.91	62.84	62.92
Boerne, TX	718	68.06	74.57	68.73
Ouray, CO	749	75.97	74.90	77.48
Big Bear Lake, CA	866	73.86	65.48	61.22
Ashland, OR	927	64.75	68.97	67.50
Boulder City, NV	976	67.33	72.95	71.55
Truckee, CA	1027	64.30	63.54	65.83
Sedona, AZ	1032	73.25	70.23	70.34
Los Gatos, CA	1082	69.73	76.65	70.14

Table 3. Probability of route failure in each city generated by the different routing methods, at a return probability of 30%.

yields distinct results, with the following observable characteristics for each approach

4.1 **Nearest Neighbor Heuristic (NNH)** The results of the NNH algorithm, shown in Figure 10, indicates that the resulting distance exhibits a direct linear relationship with the total number of nodes, with NNH showing a slope of about 110 meters/node. In terms of the probability of failure, there is high scatter, but the trend remains within the range of 60 to 80%, irrespective of city size. Finally, due to its simplicity, NNH proved to be the routing method that required the least processing time for large cities, as shown in Table 3.

4.2 **Saving Algorithm** Although the results for the Saving Algorithm method are quite similar to NNH, a closer inspection reveals that the resulting distance is slightly shorter on average, particularly in large cities (Figure 10). When considering the probability of failure, the Saving Algorithm yielded overall better results (lower failure rate)

Name of Sample City	Number of Inter-sections	Processing Time (s)		
		NNH	Saving	SA
Crested Butte, CO	98	1.15	0.58	8.88
Leavenworth, WA	113	1.33	1.09	10.23
Calistoga, CA	221	1.44	1.13	25.49
Moab, UT	239	1.86	1.51	28.76
Sausalito, CA	258	2.04	1.81	30.81
Marfa, TX	272	1.66	1.59	42.69
Carmel-by-the-Sea, CA	283	2.82	2.49	45.74
Jackson, WY	334	2.8	2.62	63.17
Whitefish, MT	319	4.85	4.04	53.04
Piedmont, CA	352	4.08	3.86	59.34
Breckenridge, CO	373	3.72	5.13	77.53
Healdsburg, CA	455	5.00	6.84	102.51
Bar Harbor, ME	603	8.92	13.85	146.92
Park City, UT	606	10.40	14.61	177.53
Taos, NM	620	10.11	14.77	180.06
Boerne, TX	718	14.60	16.55	254.13
Ouray, CO	749	18.84	25.03	244.42
Big Bear Lake, CA	866	27.77	38.18	362.97
Ashland, OR	927	34.22	48.33	513.94
Boulder City, NV	976	28.63	49.14	564.07
Truckee, CA	1027	34.81	56.57	577.27
Sedona, AZ	1032	38.64	57.46	517.80
Los Gatos, CA	1082	46.04	65.72	603.01

Table 4. Processing time of the routes in each city generated by the three different routing methods.

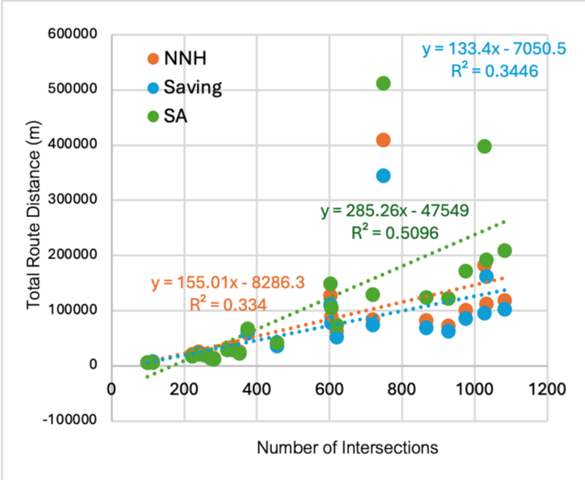

Figure 9. Total number of intersections and total route distance, with a linear trendline.

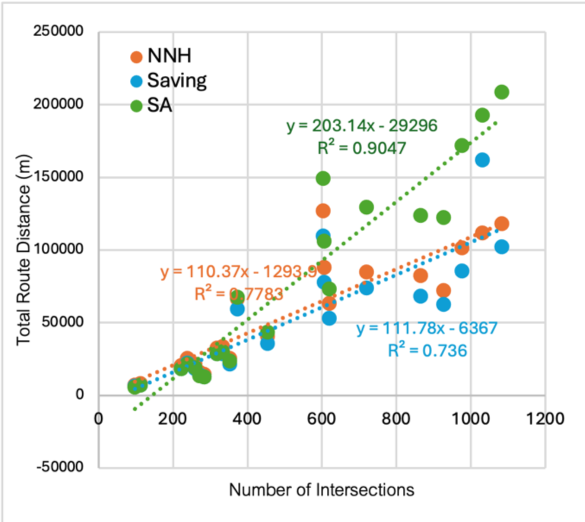

Figure 10. Total number of intersections and total route distance, with a linear trendline, excluding outliers.

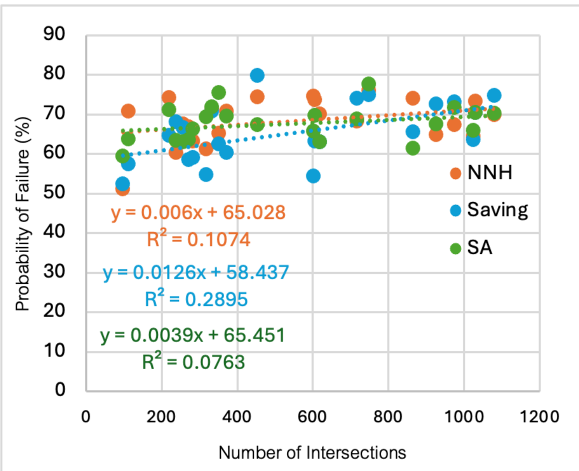

Figure 11. Total number of intersections and the probability of failure, with a linear trendline.

than the other methods, with the difference being clearly noticeable in small cities, as indicated in Table 4.

4.3 **Simulated Annealing (SA)** The results for Simulated Annealing (SA) showed a short distance for small cities, but for large cities, the trendline clearly indicates that the distance increases rapidly with increasing number of nodes, with Figure 10 showing a slope of around 200 meters/node. In the case of failure probability, SA exhibited a high degree of scatter but remained constant within the 60 to 80% range, similar to NNH. Finally, the SA method required a significantly longer processing time compared to the others, as illustrated in Table 3 and Figures 12 and 13.

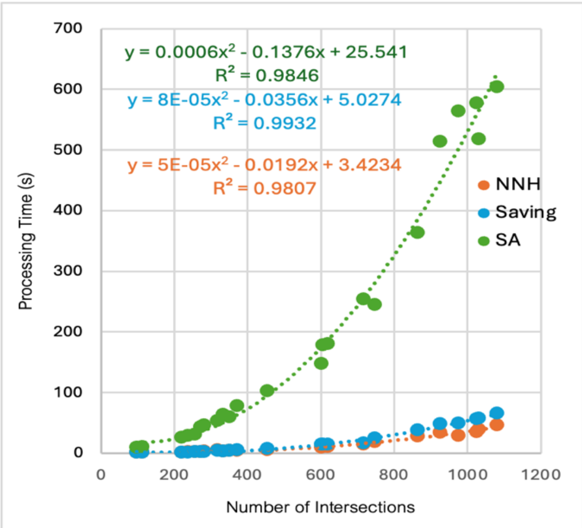

Figure 12. Total number of intersections and the processing time, with a polynomial trendline.

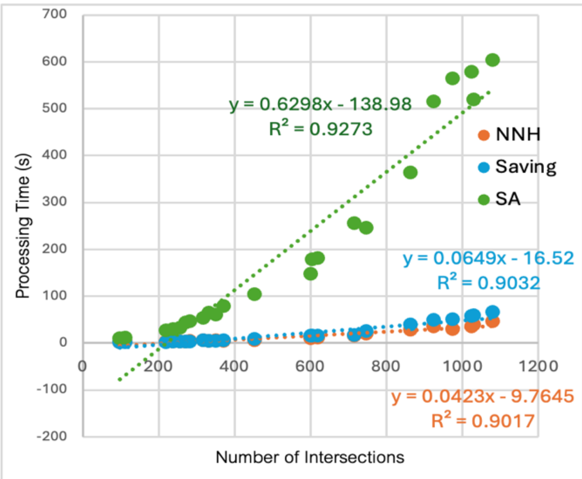

Figure 13. Total number of intersections and the processing time, with a linear trendline.

4.4 **Probability of Package Return** Figures 14-16 display the results of testing different values for the probability of package return, p. At p = 0, the probability of failure is 0%, indicating that if no returns occur, the system will not experience a failure. For values between p = 0.05 – 0.35, there is a significant increase in the probability of failure. At the higher end of this range, the rate of growth in failure probability decreases before approaching 100% failure at p = 0.40 – 0.50. The relationship between the probability of return and the probability of failure is clearly nonlinear and closely resembles a sigmoid curve.

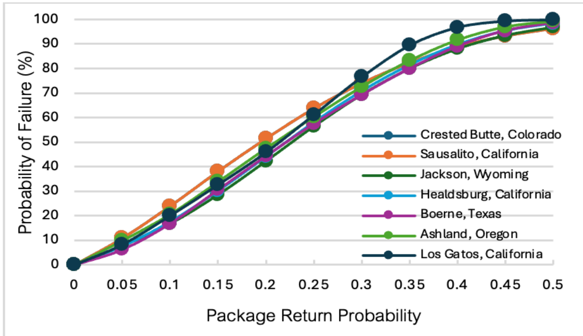

Figure 14. Package return probability and failure probability for the NNH routing method.

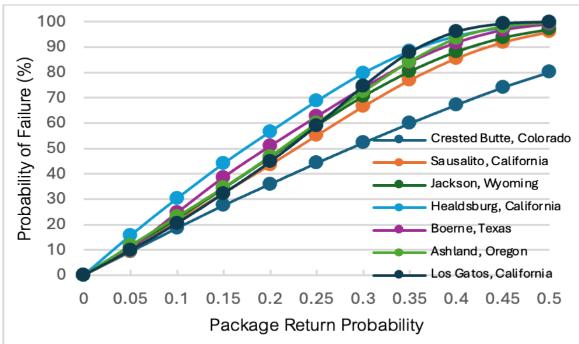

Figure 15 . Package return probability and failure probability for the Saving Algorithm routing method.

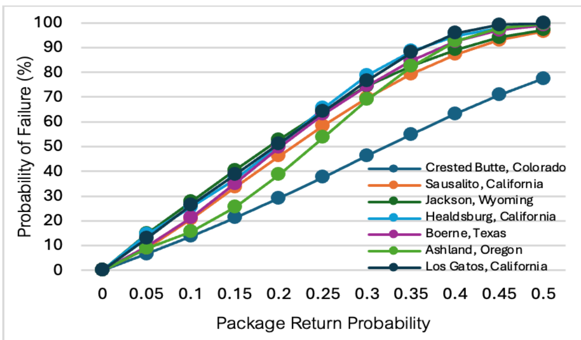

Figure 16. Package return probability and failure probability for the SA routing method.

IV. CONCLUSION

The study showed that the developed mathematical models can be effectively used to determine the optimal vehicle route for transportation when there is a probability of package returns. This outcome is explained by classifying the routing approaches. Following Liu F. et al. (2023), NNH and the Saving Algorithm fall under the same category of constructive heuristics, which build the route from scratch, leading to generally similar resulting distances. Simulated Annealing (SA), however, is a metaheuristic method, which works by iteratively improving an initially random route. Consequently, in large cities with a significant number of nodes, it is statistically more challenging for the stochastic process of SA to randomly discover a pattern that results in a shorter distance. Nevertheless, when SA is applied to smaller cities with fewer nodes, it can successfully find patterns that yield shorter distances, leading to results that are better than both NNH and the Saving Algorithm.
In terms of the probability of failure, the results showed a generally similar trend across all route generation methods. However, in small cities, the Saving Algorithm was found to have a slightly lower probability of failure compared to the other methods. Regarding processing time, NNH, being the simplest method, consumed the least time, with the Saving Algorithm requiring a comparable amount of time, while SA consistently demonstrated a trend of significantly higher processing time. In the study of varying probabilities of package return, the relationship between the probability of return and the probability of failure was found to be nonlinear and closely approximated a sigmoid curve that approaches its maximum value at $p = 0.40 - 0.50$.

It must be noted that this research did not directly solve the problem using multi-objective optimization. The routes generated by each algorithm were assessed *a posteriori* using the failure rate function through simulation. Therefore, the reduction of the failure objective (Equation 2) is an indirect result derived from the route's structure, and it was not optimized during the route construction process itself. Finally, in terms of practical application, the user needs to define what the "optimal route" means for their purpose, whether it is prioritized based on distance, probability of failure, or time required for route generation.

V. REFERENCES

Boeing, G. (2024). Modeling and Analyzing Urban Networks and Amenities with OSMnx. Working paper. https://geoffboeing.com/publications/osmnx-paper/

Chen, K.-T., Dai, Y., Fan, K., & Baba, T. (2015). A particle swarm optimization with adaptive multi-swarm strategy for capacitated vehicle routing problem. *2015 1st International Conference on Industrial Networks and Intelligent Systems (INISCom)*, 79–83. https://doi.org/10.4108/icst.iniscom.2015.258972

Clarke, G. and Wright, J.R. Scheduling of Vehicle Routing Problem from a Central Depot to a Number of Delivery Points, *Operation Research*. (1964). 12:568-581.

Cokyasar, T., Subramanyam, A., Larson, J., Stinson, M., & Sahin, O. (2022). Time-Constrained capacitated vehicle routing problem in urban E-Commerce delivery. *Transportation Research Record Journal of the Transportation Research Board*, 2677(2), 190–203. https://doi.org/10.1177/03611981221124592

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1), 80–91. https://doi.org/10.1287/mnsc.6.1.80

Desrochers, M., & Laporte, G. (1991). Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters*, 10(1), 27–36. https://doi.org/10.1016/01676377(91)90083-2

Federal Highway Administration. National Household Travel Survey, 2017. Available at https://nhts.ornl.gov, accessed on Oct. 28, 2021.

Harahap, R. F., & Sawaluddin, N. (2023). Study vehicle routing problem using Nearest Neighbor Algorithm. *Journal of Physics Conference Series*, *2421*(1), 012027. https://doi.org/10.1088/1742-6596/2421/1/012027

Liu, F. et al. (2023, March 1). Heuristics for Vehicle Routing Problem:A survey and recent advances. arXiv.org. https://arxiv.org/abs/2303.04147

Loqate. (2021). *Fixing failed deliveries: Why accurate address data is key*. https://info.loqate.com/hubfs/Loqate_Fixing%20failed%20deliveries.pdf

OpenStreetMap. (n.d.). OpenStreetMap. https://www.openstreetmap.org/about

Paessens, H. (1988). The savings algorithm for the vehicle routing problem. *European Journal of Operational Research*, 34(3), 336–344. https://doi.org/10.1016/0377-2217(88)90154-3

Van Laarhoven, P. J. M., & Aarts, E. H. L. (1987). Simulated annealing. *Springer eBooks* (pp. 7–15). https://doi.org/10.1007/978-94-015-7744-1_2

Wei, L., Zhang, Z., Zhang, D., & Leung, S. C. (2017). A simulated annealing algorithm for the capacitated vehicle routing problem with two-dimensional loading constraints. *European Journal of Operational*

Numfon Papun, & Phatthararit Kaewpradit. (2020). *The Arrangement of Goods Transportation Routes using the Saving Algorithm: A Case Study of a Plastic Pellet Factory* [Bachelor of Engineering Thesis]. Dhurakij Pundit University. (In Thai)

Acknowledgements

This research was successfully completed thanks to the generous support of Dr. Monsikarn Jansang, who devoted valuable time to the researchers, providing consultation, knowledge, and meticulous review and corrections of the manuscript. The researchers wish to express their deepest gratitude.