mathematics

Article

An Efficient Simulated Annealing Algorithm for the Vehicle Routing Problem in Omnichannel Distribution

Vincent F. Yu [1], Ching-Hsuan Lin [1], Renan S. Maglasang [1], Shih-Wei Lin [2,3,4,*] and Kuan-Fu Chen [3,5]

1 Department of Industrial Management, National Taiwan University of Science and Technology, Taipei 106, Taiwan; vincent@mail.ntust.edu.tw (V.F.Y.); m10901001@mail.ntust.edu.tw (C.-H.L.); renan@mail.ntust.edu.tw (R.S.M.)
2 Department of Information Management, Chang Gung University, Taoyuan 333, Taiwan
3 Department of Emergency Medicine, Keelung Chang Gung Memorial Hospital, Keelung 204, Taiwan; kfchen@mail.cgu.edu.tw
4 Department of Industrial Engineering and Management, Ming Chi University of Technology, New Taipei 243, Taiwan
5 College of Intelligent Computing, Chang Gung University, Taoyuan 333, Taiwan
* Correspondence: swlin@mail.cgu.edu.tw

Abstract: A variant of the vehicle routing problem (VRP) known as the Vehicle Routing Problem in Omnichannel Retailing Distribution Systems (VRPO) has recently been introduced in the literature, driven by the increasing adoption of omnichannel logistics in practice. The VRPO scenario involves a large retailer managing several stores, a depot, and a homogenous fleet of vehicles to meet the demands of both stores and online customers. This variant falls within the class of VRPs that consider precedence constraints. Although the vehicle routing problem in omnichannel retailing distribution (VRPO) has been addressed using a few heuristic and metaheuristic approaches, the use of Simulated Annealing (SA) remains largely unexplored in the pickup and delivery problem (PDP) literature, both before and after the rise of omnichannel logistics. This article introduces the Efficient Simulated Annealing (ESA) algorithm, demonstrating its suitability in generating new benchmark solutions for the VRPO. In experiments with sixty large instances, ESA significantly outperformed two previous algorithms, discovering new best-known solutions (BKSs) in fifty-nine out of sixty cases. Additionally, ESA demonstrated superior efficiency in 68.3% of the test cases in terms of reduced computational times, showcasing its higher effectiveness in handling complex VRPO instances

Keywords: vehicle routing problem; pickup and delivery; simulated annealing; metaheuristics; omnichannel distribution

MSC: 90C27

Citation: Yu, V.F.; Lin, C.-H.; Maglasang, R.S.; Lin, S.-W.; Chen, K.-F. An Efficient Simulated Annealing Algorithm for the Vehicle Routing Problem in Omnichannel Distribution. *Mathematics* **2024**, *12*, 3664. https://doi.org/10.3390/math12233664

Academic Editor: Paolo Mercorelli

Received: 17 October 2024
Revised: 14 November 2024
Accepted: 19 November 2024
Published: 22 November 2024

1. Introduction

The progress in omnichannel operations gives rise to increasingly complex last-mile distribution networks over which customer information flows quickly and timely access is critical for effective customer response. As the last mile represents the most costly segment of the supply chain, achieving a cost-efficient vehicle routing continues to be a significant challenge. One of the most-studied Vehicle Routing Problem (VRP) variants arising from omnichannel applications is the Pickup and Delivery Problem (PDP). The PDP uses a set of vehicles to fulfill transportation requests consisting of a pickup location and a delivery location with a precedence constraint. Lokin [1] initially explored PDP, introducing a modified version of the traveling salesman problem. Precedence relations are imposed on specific customers, requiring certain nodes to be visited before others. The

PDP consists of three main types of problems based on the origin–destination of each request, namely, many-to-many problems (type 1), one-to-many-to-one problems (type 2), and one-to-one problems (type 3) [2]. In type 1 PDP, requests are carried from more than one start node to more than one end node. In type 2 PDP, some requests start at the depot and end at many customers, while others start at a customer and end at the depot. In type 3 PDPs, each request moves from a single start node to a single end node.

In this work, we build on the capacitated Vehicle Routing Problem in Omnichannel Retailing Distribution Systems (VRPO), which was first introduced by Abdulkader et al. [3]. The VRPO is a VRP hybrid with CVRP and PDP features. It is an instance of omnichannel distribution, which is a business model that seeks to offer customers a seamless experience that involves both physical and online stores. The VRPO has two types of requests: (i) requests for store pickup and/or delivery and (ii) requests for customer delivery. Customer requests are made through online order points. A VRPO route follows the depot–store–customer precedence. This means that once a vehicle departs from the depot, it should pick up all store requests assigned to it before visiting the first store (delivery). Furthermore, the same vehicle should load (pickup) from the same store all the customer requests assigned to it before visiting those customers (delivery).

The VRPO inherits the NP-hard complexity of the VRP, resulting in prohibitively high computational time to reach an optimal solution. As exact methods struggle with scalability, heuristic approaches become a practical alternative. However, despite advances in heuristic approaches, a gap exists in the application of Simulated Annealing within the omnichannel VRP context. The reviewed studies predominantly rely on genetic algorithms, savings heuristics, and hybrid approaches, leaving SA underexplored. This gap highlights the potential for future research to utilize SA, leveraging its strengths in navigating complex solution landscapes and escaping local optima, to address the challenges presented by omnichannel logistics. Given the growing attention to the challenges of omnichannel VRPs and the clear literature gap in exploring simulated annealing (SA) for PDP, this work focuses on tailoring an SA-based metaheuristic approach, the Efficient Simulated Annealing (ESA) for the VRPO.

The remainder of this paper is organized as follows: Section 2 discusses the related works. Section 3 details the ESA for the VRPO. Section 4 presents the computational experiment and analysis. Finally, Section 5 discusses the conclusions and recommendations for future studies.

2. Review of Related Literature

2.1. The Pickup and Delivery Problem in Vehicle Routing

The VRP is a combinatorial optimization problem that aims to determine the optimal routes for a fleet of vehicles to deliver orders to customers while minimizing costs [4]. The PDP, a key VRP class, aims to determine the most efficient routes for a fleet of capacitated vehicles to fulfill customer requests, each involving a pickup location and a corresponding delivery location. Vehicles start from the depot, pick up or deliver goods at specified locations, and return to the depot. The mathematical formulation of the general PDP, with its variants classified, is given by Parragh et al. [5]. The PDP has been extensively studied by Battara et al. [6] and divided into three categories: (i) many-to-many, in which commodities can have multiple origins and destinations, (ii) one-to-many-to-one, in which some commodities are delivered from a depot to customers and others are collected from customers and returned to the depot, and (iii) one-to-one, in which each commodity has a single origin and destination. Parragh et al. [5,7] provided another classification of PDP into two general classes: (i) transportation to/from the depot and (ii) transportation between customers. The first class includes cases in which deliveries (linehauls) must occur before pickups (backhauls), linehauls and backhauls can be served in any order, and customers have both linehaul and backhaul needs, either requiring or not requiring a single visit. The second class involves unpaired pickup and delivery locations, in which any pickup can fulfill any delivery, and paired locations, in which no other customer is visited between pickup and delivery. Since then, the development of the PDP has been extensive, but the recent rise of omnichannel distribution has introduced new challenges that add complexity to the problem.

2.2. VRP in Omnichannel Distribution Context

The omnichannel logistics landscape has evolved significantly in recent years, driven by the increasing demand for seamless integration across multiple sales and distribution channels, including physical stores, online platforms, and hybrid modes. In omnichannel, the interactions of customers with the retailer can be varied via online or physical stores. This complexity poses significant challenges for the omnichannel VRP giving way to the class of omnichannel VRP involving pickups, or deliveries, or mixed, or simultaneous. In VRPO, the fleet of vehicles is required to simultaneously perform bulk deliveries to retail stores from a main central depot and pick up online customer orders from retail stores and subsequently deliver orders to online customers.

The VRPO can be considered as an integrated 2E-VRP since retailers and online consumers correspond to the customers of two distinct layers of the supply chain. A categorization of multiple-channel retailing in multi-, cross-, and omnichannel retailing for retailers and retailing is comprehensively presented by Beck and Rygl [8]. The VRPO involves two types of customers: some with pickup requests, some with delivery requests, or both. A customer with both types of requests is visited once, and a pickup can be paired with multiple deliveries. The network consists of a depot or a distribution center, some local stores, and some home-delivery consumers. Vehicles serve both stores (for stock replenishment) and home delivery customers using the same fleet. A key difference in the VRPO is that only stores handle both pickups and deliveries, while consumers receive deliveries only. Additionally, vehicles follow a depot–store–customer visit order, loading all store requests from the depot before making deliveries. Customer orders are placed through multiple online platforms, such as apps and websites. For a detailed mathematical formulation of the general PDP, readers can refer to Abdulkader et al. [3].

Table 1 outlines the recent studies on omnichannel vehicle routing problems. Since the combination of VRP and PDP in an omnichannel retail context was first introduced in year 2018, we have limited our review to related papers published from 2018 to the present. The related literature on VRP under omnichannel logistics encompasses a variety of problem classes and solution methodologies, illustrating the increasing complexity of modern last-mile distribution. Early works such as those by Martins et al. [9], Martins et al. [10], and Bayliss et al. [11] focused on mixed and simultaneous pickup and delivery (MPD and SPD) problems. They tackled the dual challenges of replenishing retail stores while managing direct-to-consumer deliveries, emphasizing operational efficiency through mixed routes and shared fleets. Sawicki et al. [12] further extended this focus by exploring a two-tier system that serves both retail stores and online customers, while Janjevic et al. [13] introduced a three-tier, multi-modal last-mile network with differentiated service levels, integrating city-level and regional deliveries. Guerrero-Lorente et al. [14] provided a novel approach by incorporating both forward and reverse logistics in omnichannel VRP, accounting for both deliveries and returns, which added complexity to managing customer preferences.

More recently, Liu et al. [15] addressed MPD problem that aimed to balance cost minimization and customer satisfaction. Hendalianpour et al. [16] contributed to the literature by focusing on multi-product and multi-level VRP, managing replenishments across different levels in an omnichannel distribution network. Li et al. [17] introduced the selective many-to-many pickup and delivery problem with handling costs (SMMPDPH), which extended traditional VRP by addressing selective routing among multiple nodes. This work highlighted the importance of managing handling costs, especially when goods are being loaded or unloaded at multiple locations. The most recent studies by Yang and Li [18], Qiu et al. [19], and Li and Wang [20] focused on inventory replenishment, strict time windows, and multicommodity deliveries in omnichannel settings, reflecting the growing demand for efficient logistics solutions that meet diverse customer expectations.

From a solution approach perspective, the reviewed studies illustrate the diverse strategies used to tackle these omnichannel VRP. Martins et al. [9] and Martins et al. [10] used simheuristics and savings-based heuristics, respectively, while Bayliss et al. [11] employed a discrete-event heuristic followed by a local search, demonstrating hybrid approaches for MPD. Guerrero-Lorente et al. [14] and Janjevic et al. [13] leveraged mixed-integer programming (MIP) combined with continuous approximation and capacitated location routing to optimize complex network configurations. Li et al. [17] utilized mixed-integer programming (MIP) alongside iterated local search (ILS) and a memetic algorithm (MA) to solve the SMMPDPH. Liu et al. [15] applied advanced metaheuristics, including MOGWO and NSGA-II, while Hendalianpour et al. [16] employed Benders decomposition and Lagrangian relaxation to solve multi-level problems. In their recent work, Qiu et al. [19] proposed an optimization model focusing on capacity-sharing, and Li and Wang [20] utilized adaptive large neighborhood search (ALNS) to manage time-sensitive deliveries. Yang and Li [18] employed a hybrid heuristic to efficiently handle multicommodity flows in an omnichannel context. Finally, Abdulkader et al. [3], our baseline study, proposed two algorithms for comparison, the two-phased heuristic (2PH) and multi-ant colony optimization (MAC).

In pre-omnichannel PDP applications, Wang et al. used a parallel simulated annealing (p-SA) algorithm to solve VRPSPDs with time windows, comparing results to existing benchmarks [21]. Avci and Topaloglu developed a hybrid local search algorithm combining simulated annealing and variable neighborhood descent for VRPSPD and VRPMPD, with an adaptive self-tuning threshold function [22]. Danloup et al. used large neighborhood search (LNS) and genetic algorithms (GA) to solve SPD with transshipment, comparing results to other algorithms [23]. Koc et al. reviewed the VRPSPD, focusing on future directions but lacking insights into metaheuristics [24]. While genetic algorithms (GA) and tabu search (TS) are well-studied, SA remains understudied in PDP and VRPSPD domains. Additionally, Parragh et al. [6,8] highlighted that genetic algorithms, tabu search and simulated annealing are among the most commonly used metaheuristics, with SA being the least explored of the three.

Table 1. Recent studies in omnichannel VRP context.

Year	Authors	VRP Class	Problem Attributes	Solution Approach
2020	Martins et al. [9]	PD	Handles retail store replenishment and customer deliveries using a shared fleet for both tasks.	Savings-Based Heuristic
2020	Martins et al. [10]	SPD	Replenishes retail stores and serves direct home deliveries with stochastic travel times.	Simheuristic and Biased-Randomized Heuristic
2020	Bayliss et al. [11]	MPD	Integrates retailer replenishment and direct home deliveries; combines store replenishment with customer-focused routes.	Discrete-Event Heuristic and Local Search
2020	Guerrero-Lorente et al. [14]	PD	Manages deliveries and returns across a multi-facility network; supports customer preference with city distribution centers and parcel stations.	Mixed-Integer Programming and Continuous Approximation
2021	Sawicki et al. [12]	Multi-echelon VRP with Direct Delivery	Serves retail stores and online customers through a two-tier distribution system; focuses on optimizing direct deliveries from central depots and intermediate hubs.	Mixed-integer programming

2021	Janjevic et al. [13]	Multi-Tier, Multi-Service Level VRP	Replenishes multiple levels of retail inventory; involves replenishing retail stores and satellite hubs, as well as delivering products directly to customers.	Capacitated Location-Routing and Continuum Approximation
2022	Liu et al. [15]	MPD	Serves retail stores and home customers; integrates replenishment of stores with direct customer deliveries, aiming for cost minimization and customer satisfaction.	Multi-Objective Gray Wolf Optimizer (MOGWO) and Non-dominated Sorting Genetic Algorithm II (NSGA-II)
2022	Hendalianpour et al. [16]	Multi-product, Multi-level Omnichannel VRP	Replenishes multiple levels of retail inventory; involves replenishing retail stores and satellite hubs, as well as delivering products directly to customers.	Benders Decomposition and Lagrangian Relaxation
2022	Li et al. [17]	Selective Many-to-Many Pickup and Delivery (SMMPDPH)	Manages selective pickup and delivery among multiple nodes; focuses on minimizing handling costs in omnichannel last-mile delivery.	Mixed-Integer Programming, Iterated Local Search (ILS), and Memetic Algorithm (MA)
2023	Yang and Li [18]	PD	Performs pickup and delivery between distribution centers, retail stores, and end customers; focuses on efficient transport of heterogeneous goods in omnichannel retail.	Mixed-Integer Programming, Tabu Thresholding and Memetic Algorithms
2025	Li and Wang [20]	PD	Replenishes retail stores and satellites; manages split deliveries under strict time window constraints for various product types.	Adaptive Large Neighborhood Search (ALNS)
2025	Qiu et al. [19]	Inventory Replenishment VRP with Capacity-Sharing	Replenishes inventory across retail stores; uses a capacity-sharing strategy to reduce travel and inventory holding costs while meeting online and in-store demands.	Optimization Model and Solution Procedure

This paper addresses the gap by applying SA in extensive computational experiments for VRP cases. Since its introduction by Kirkpatrick et al. [25], SA has been widely applied, extended, and combined with other methods to solve complex optimization problems. SA's strength lies in avoiding local optima, making it well-suited for combinatorial problems like the VRPO when properly adjusted. While there are no studies focusing on simulated annealing in the post-omnichannel problem context, some newly tweaked SA algorithms include an improved simulated annealing algorithm with a crossover operator for the capacitated vehicle routing problem [26]; multiple-temperature simulated annealing for the permutation flowshop scheduling problem [27]; SA with variable neighborhood descent for the heterogeneous fleet vehicle routing problem [28]; a fast simulated annealing algorithm for the examination timetabling problem [29]; a new hyper-heuristic based on adaptive simulated annealing and reinforcement learning for the capacitated electric vehicle routing problem [30]; SA with a mutation strategy for the share-a-ride problem with flexible compartments [31]; multi-start simulated annealing for the team-orienteering problem [32]; a backtracking SA metaheuristic for the job-shop scheduling problem [33]; a chaos-enhanced SA for hybrid flowshop scheduling with identical machines [34]; and a simulated annealing algorithm for the vehicle routing problem with parcel lockers [35].

3. Solution Approach

This paper presents an efficient SA-based algorithm that employs a method to predetermine the ranking of parameters, namely, T_0, N_{iter}, *beta*, N_{nimp}, and *UnitP*. The first three are standard in classical SA, while the last two are efficiency parameters. The T_0, N_{iter}, and *beta* control the temperature cooling, while N_{nimp} and *UnitP* manage solution acceptance and stopping policy. N_{imp}, the "non-improving solution" parameter, terminates the outer loop if no improvement is found after a set number of iterations. *UnitP* is used in the penalty method to speed up the search by avoiding redundant objective function evaluations. The components of our ESA are detailed in the next sections.

3.1. Solution Encoding

The VRPO solution array includes *c* customers {C_1, C_2, …, C_C}, *r* stores {R_1, R_2, …, R_r}, and *n* dummy zeros representing the start and end of routes. In Figure 1a–c, a sample solution is shown as a single array with one dummy zero, three stores, and eighteen customers, where dummy zeros act as route separators. The array is decoded into two routes. Route 1 starts at the depot, visits store R_2 for stock replenishment and to pick up customer requests, delivers to customers C_{11}, C_{10}, C_{18}, C_{12}, C_{17}, C_6, C_9, and C_5, and returns to the depot. Route 2 visits R_3 for customer pickups, delivers to customers C_1 and C_3, replenishes stock at store R_1, picks up customer requests, delivers to customers C_2, C_{15}, C_{16}, C_{13}, C_7, C_4, C_{14}, and C_8, and returns to the depot.

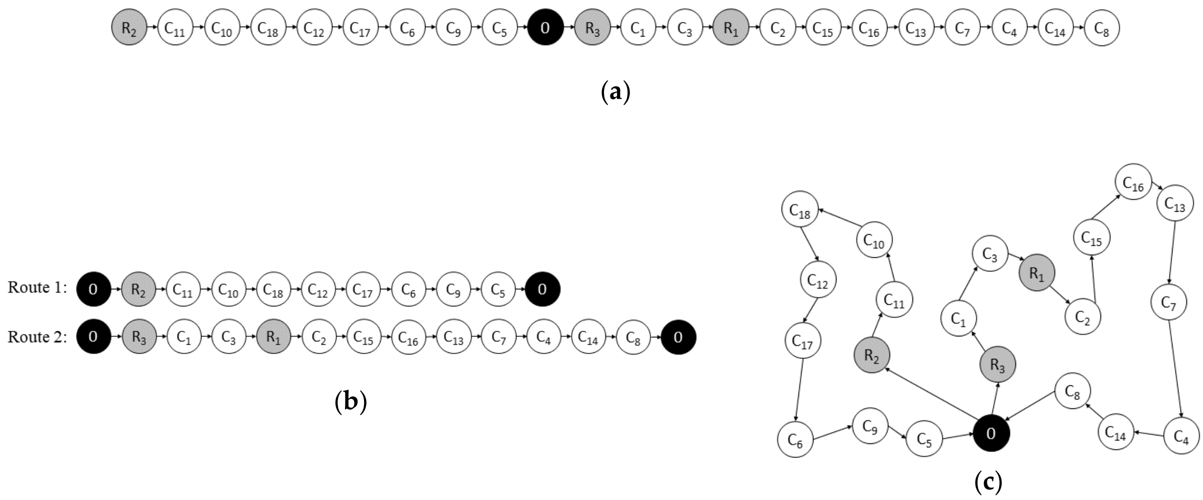


Figure 1. VRPO solution representation: (**a**) array representation of a sample solution; (**b**) vehicle route representation; (**c**) graphical representation.

Each element of the solution array is decoded from left to right to form the routes. There are three cases for handling dummy zeros, as follows:

Case 1: If the next element after a dummy zero is another dummy zero, it is ignored, and the process continues.

Case 2: If the next element is a local store, the current vehicle terminates its route and returns to the warehouse, a new vehicle is activated, and decoding proceeds.

Case 3: If the next element is a customer, it is ignored, and decoding continues.

After decoding, travel time and vehicle load are evaluated, ensuring store and customer demands are met. If not, a penalty is applied based on the degree of violation, including for exceeding the maximum tour length.

3.2. Initialization

The initial solution is constructed using a simple rule to form a complete solution array; one vehicle is activated per local store. After visiting a store, the vehicle serves the closest unvisited customer. Algorithm 1 illustrates the detailed initialization procedure.

```
Algorithm 1: ESA Initialization for the VRPO
Function: Initialize to obtain an initial solution;
  Do
    A vehicle departs from the depot;
    A vehicle travels to the closest unused local store;
    Update: (1) vehicle's current location, (2) vehicle load;
    Do
        Find the closest unserved customer which satisfies the following conditions:
        (1) The current vehicle can provide enough goods for this customer;
        (2) If this customer is served and the current vehicle
            goes back to the warehouse, it will not violate the max time of the vehicle;
            If (one customer is found) {
                The current vehicle goes to serve this customer; and
                Update current vehicle location and the amount of goods in the vehicle;
                Update the partial solution;
                }
    } Until no unserved customer can be added to the current vehicle route;
        The current vehicle goes back to the depot;
        Update the partial solution;
  } Until all local stores are used
```

3.3. Neighborhood Operators: Swap, Insert, and Invert

To explore neighborhood solutions, the current solution S is perturbed using three neighborhood moves: insert, swap, and invert. A new solution S', is generated at each iteration by applying one of these moves. The *Insert* move selects a random element of S and places it before another random position. The *Swap* move selects two random elements in S and swaps them. The *Invert* move selects a random subsequence of S and reverses its order. The probabilities for choosing each move are set equally.

3.4. Efficient Simulated Annealing (ESA) Algorithm

Algorithm 2 outlines the ESA algorithm. T_0 is the initial temperature, N_{iter} is the number of iterations at each temperature, and N_{nimp} is the maximum number of consecutive iterations without improvement. This parameter represents the maximum number of temperature reductions without finding a better solution. *UnitP* is the penalty applied for constraint violations, and *beta* is the cooling schedule coefficient. The algorithm proceeds as described.

The current temperature (T) is set to the initial temperature (T_0), and the best solution (S_{best}) is initialized as the current solution (S). The best objective function value (OFV_{best}) is set as $OF(S_{\text{best}}, UnitP)$, where OF is the objective function. A penalty, which is calculated as the unit penalty times the degree of the constraint violation, is applied when (1) the store product quantity is insufficient for the route, (2) the vehicle load exceeds capacity, or (3) the tour length exceeds the limit.

The search goes through a neighborhood search mechanism for each iteration at a given temperature. Selection probabilities, set uniformly at one-third for each of the three move operators, determine the move. Let *delta* be the difference between the new solution (S') and the current solution (S), that is, *delta* = $OF(S', UnitP)$-$OF(S, UnitP)$. If *delta* <= 0, S' replaces S; otherwise, S' is accepted at a specific probability *lambda* between 0 and 1. If *lambda* is less than or equal to $e^{-delta/T}$, S' replaces S. The current temperature drops to T (*beta*), where 0 < *beta* < 1 after N_{iter} iterations at the current temperature. The search terminates when the best solution has not improved after N_{nimp} consecutive temperature reductions. Also, the best solution (S_{best}) and its objective function value (OFV_{best}) are updated when a better feasible solution is found. When the algorithm terminates, S_{best} and OFV_{best} are returned.

Algorithm 2: ESA for the VRPO

```
Function: ESA-VRPO to find the best solution, S_best;
Inputs: T0, N_iter, N_nimp, UnitP, beta, and "problem instances";
begin
    Generate initial solution, S ← Initialize;
      T ← T0, S_best ← S, N_count ← 0, OFV_best ← OF(S, UnitP);
      While (nimp < N_nimp) do
          For iter = 1 to N_iter do
              Generate prob = uniform (0,1);
              If (prob <= ⅓) ← Swap;
              If (⅓ < prob <= ⅔) ← Insert;
              If (⅔ < prob <= 1) ← Invert;
              delta = OF(S′, UnitP)-OF(S, UnitP);
              If (delta <= 0) then
                  S ← S′;
              else
                  Generate lambda = uniform (0,1);
                  If (lambda <= e^(-delta/T)) then
                      S ← S′;
                  endif
              endif
              If (S′ is feasible and OF(S′, UnitP) <= OFV_best) then
                  S_best ← S′, OFV_best = OF(S′, UnitP),  nimp ← 0;
              endif
          endfor
          nimp = nimp +1; T ← T(beta);
      endwhile
      return S_best, OFV_best;
end
```

4. Experiment

4.1. Test Instances

The small and large VRPO instance sets, which were used as benchmarks for this study, were obtained from Abdulkader et al. [3] as benchmarks for the comparative study. For details on data generation, readers may refer to their work. The small set includes 20 instances with 3, 4, 5, or 6 retail stores and 6, 9, 12, 15, or 18 consumers. The large set contains 60 instances with 10, 15, 20, or 25 stores and 25, 50, 75, 100, or 150 consumers, across three inventory cases (see Table 2). Instance data include the depot, stores, consumers, store products, stock levels, x-y coordinates, and pickup/delivery requests.

Table 2. Inventory availability cases.

Case	Stock Availability	Description
(1) *Low*	$\sum I_p = \sum D_p + U[0.1, 0.2] * \sum D_p$	Total stock in the network exceeds consumer demand by 10–20%.
(2) *Moderate*	$\sum I_p = \sum D_p + U[0.5, 1.0] * \sum D_p$	Inventory exceeds demand by 50–100%.
(3) *High*	$\sum I_p = \sum D_p$	Each store's stock can fully meet all demands across the network.

4.2. Parameter Setting and Fine-Tuning

The algorithm-related parameters are listed and defined in Table 3, where L is the length of the solution array. With the five factors at four levels each, a total of 1024 experiments should be conducted. Alternatively, a representative sampling was conducted using Type B of the Taguchi L16 (4^5) orthogonal experimental design. A total of sixteen experimental combinations were used and compared. For each combination, ESA is implemented ten times independently per random instance of six instances.

Table 3. ESA parameter levels.

Parameter	Definition	Levels
T_0	Initial temperature	10, 20, 30, 40
N_{iter}	Number of iterations per T level	2000 L, 2500 L, 3000 L, 3500 L
beta	Cooling coefficient	0.90, 0.93, 0.96, 0.99
N_{nimp}	Maximum iterations over non-improving solutions	20, 30, 40, 50
UnitP	Unit penalty per constraint violation	1500, 2000, 2500, 3000

The performance of each combination is measured using the average relative percentage deviation (%Gap) and average computational time from 10 independent runs per combination. Tables 4 and 5 present the results. Table 4 shows the average %Gap and computational times obtained from the predetermined parameter levels. The results highlight combinations 4 and 13 on Niter's effect on the %Gap. The former has the least average %Gap (i.e., the least deviation with respect to the best solution found) at the highest Niter, while the latter has the highest average %Gap at the least Niter.

Table 5 shows the rankings of parameters based on percentage gaps. It reveals Niter as the most critical parameter with the largest average %Gap range. Parameters *beta* and T_0 are the second and third in the ranking, respectively. It is hard to separate the influence of *beta* and T_0 when determining the current T in the algorithm. T affects the probability of accepting worse solutions. Generally, a worse solution is frequently accepted at a higher T (i.e., the earlier search stage). As a result, the convergence at this stage is unlikely. As T decreases (i.e., the later search stage), worse solutions are often rejected, and the search gravitates locally. Lesser *beta* values cause drastic drops of current T and often cause the search process to become prematurely stuck in local optima. N_{nimp} helps in termination and convergence. Higher values of N_{nimp} may lead to more and unnecessary evaluations during the search. *UnitP* is considered a trivial parameter, as the search is most likely to reject an infeasible solution, let alone one with a high *UnitP*. Finally, to balance solution quality and computation time, the algorithm's T_0, N_{iter}, *beta*, N_{nimp}, and *UnitP* parameter values were set to 30, 3500L, 0.96, 50, and 1500, respectively.

Table 4. CPU times and %Gaps of predetermined parameter levels.

Combinations	T_0	N_{iter}	*beta*	N_{nimp}	*UnitP*	%Gap * (Ave.)	Time (Ave.)
1	10	2000 L	0.90	20	1500	8.561	28.87
2	10	2500 L	0.93	30	2000	7.524	49.28
3	10	3000 L	0.96	40	2500	6.161	91.06
4	10	3500 L	0.99	50	3000	4.164	279.94
5	20	2000 L	0.93	40	3000	7.191	49.56
6	20	2500 L	0.90	50	2500	7.548	57.93
7	20	3000 L	0.99	20	2000	6.810	202.49
8	20	3500 L	0.96	30	1500	5.060	109.09
9	30	2000 L	0.96	50	2000	5.847	79.33
10	30	2500 L	0.99	40	1500	4.795	252.13
11	30	3000 L	0.90	30	3000	6.682	54.22
12	30	3500 L	0.93	20	2500	5.634	68.91
13	40	2000 L	0.99	30	2500	8.863	176.05
14	40	2500 L	0.96	20	3000	5.803	78.35
15	40	3000 L	0.93	50	1500	6.116	85.33
16	40	3500 L	0.90	40	2000	6.155	72.60

* $\%Gap^i = \frac{Obj^i_{ave} - Obj_{best}}{Obj_{best} x\ 100\%}$. Obj^i_{ave} denotes the average objective function value obtained in 10 runs per combination. Obj_{best} is the instance's minimum objective function value obtained in 200 runs.

Table 5. Rankings of parameters based on %Gap.

Levels	%Gap (Ave.)					Time (Ave.)				
	T_0	N_{iter}	*beta*	N_{nimp}	*UnitP*	T_0	N_{iter}	*beta*	N_{nimp}	*UnitP*
1	6.6025	7.6154	7.2365	6.7021	6.1330	107.26	81.71	52.46	92.06	114.50
2	6.6522	6.4177	6.6164	7.0323	6.5843	100.09	105.42	62.18	95.40	97.22
3	5.7398	6.4424	5.7177	6.0756	7.0516	109.94	104.98	86.35	111.95	96.84
4	6.7344	5.2535	6.1583	5.9189	5.9601	103.08	128.27	219.39	120.97	111.82
Range	0.9946	2.3619	1.5188	1.1134	1.0915	9.85	46.56	166.93	28.90	17.66
Rank	5	1	2	3	4	5	2	1	3	4

4.3. Results and Discussion

This section presents the results of ESA's performance compared to the baseline algorithms 2PH and MAC. The experiment was implemented on a server with four 2.1 GHz processors with 16 cores each and 256 GB RAM. Our implementation used Microsoft Visual C++ 2019 on a desktop with an Intel Core i9-10900 CPU @ 2.80GHz, 64 GB of RAM, and Windows 11.

4.3.1. Small Instances

In small problems, the results as shown in Table 6 suggest that our ESA is at par in terms of solution quality and average solution quality but generally superior in terms of computational time. ESA consistently obtains the optimal solution for all twenty small instances, as demonstrated in the ESA (Best) and %Gap (Best) columns of Table 6, matching the optimal solution provided by the exact solver. While our results are clearly better than those of 2PH, there are slight differences in the five instances in which our results are compared to MAC. Finally, ESA outperforms MAC in three instances a8, a15, and a20. However, ESA is outperformed by MAC in two instances: a5 and a10.

Table 6. Comparison of ESA to 2PH, and MAC in small instances.

	MINLP by Gurobi		**Abdulkader et al.** [3]				**ESA**				
Ins	**Optimal**	**Time (sec.)**	**2PH**	**%Gap**	**MAC**	**%Gap**	**ESA (Best)**	**ESA (Ave.)**	**Time (Ave.)**	**%Gap * (Best)**	**%Gap * (Ave.)**
a1	386.906	0.11	479.19	23.85	386.91	0.00	386.906	386.906	1.00	0.00	0.00
a2	416.346	1.31	589.63	41.62	416.35	0.00	416.346	416.346	1.50	0.00	0.00
a3	424.309	5.27	512.32	20.74	424.31	0.00	424.309	424.309	2.40	0.00	0.00
a4	455.288	28.70	767.06	68.48	455.29	0.00	455.288	455.288	3.80	0.00	0.00
a5	601.363	179.55	936.20	55.68	601.36	0.00	601.363	702.640	4.90	0.00	16.84
a6	419.327	2.16	512.12	22.13	419.32	0.00	419.327	419.327	1.20	0.00	0.00
a7	455.912	14.66	688.39	50.99	455.91	0.00	455.912	455.912	1.80	0.00	0.00
a8	448.831	153.45	711.89	58.61	449.35	0.12	448.831	448.831	2.80	0.00	0.00
a9	457.319	8.31	746.57	63.25	457.32	0.00	457.319	457.319	4.20	0.00	0.00
a10	514.376	607.81	740.80	44.02	514.38	0.00	514.376	518.290	5.40	0.00	0.76
a11	486.037	4.70	545.72	12.28	486.04	0.00	486.037	486.037	1.40	0.00	0.00
a12	624.812	48.09	881.95	41.15	624.81	0.00	624.812	624.812	1.90	0.00	0.00
a13	535.437	139.53	961.67	79.60	535.44	0.00	535.437	535.437	3.10	0.00	0.00
a14	605.028	166.98	838.93	38.66	605.03	0.00	605.028	605.028	4.40	0.00	0.00
a15	708.226	601.25	898.75	26.90	709.89	0.23	708.226	708.890	6.30	0.00	0.09
a16	468.885	6.30	582.10	24.15	468.89	0.00	468.885	468.885	1.60	0.00	0.00
a17	468.610	139.83	608.99	29.96	468.61	0.00	468.610	468.610	2.70	0.00	0.00
a18	586.624	14400.00	924.41	57.58	586.62	0.00	586.624	586.624	3.70	0.00	0.00
a19	750.937	437.11	964.11	28.39	750.94	0.00	750.937	750.937	5.10	0.00	0.00
a20	588.654	14400.00	1005.72	70.85	601.71	2.22	588.654	599.751	6.90	0.00	1.89

* $\%Gap = \frac{obj_{Heuristic} - obj_{MILP}}{obj_{MILP}} \times 100\%$.

4.3.2. Large Instances

The large instances feature up to 25 local stores and 150 customers. To further evaluate these instances, three inventory scenarios are considered. Table 7a–c show the results for instances b1–b20 under *low* inventory, b21–b40 under *moderate* inventory, and b41–b60 under *high* inventory.

1. Best Know Solutions (BKS): The ESA found fifty-nine new BKS out of sixty large problem instances compared to the 2PH and MAC results. However, for instance b31, MAC's result remains the BKS (refer to the BKS column).
2. Performance Comparison Based on BKS: Using the BKS, we calculated the percentage gaps for 2PH and MAC, which are shown in the %Gap (2PH) and %Gap (MAC) columns. Table 7d summarizes the performance improvements of ESA over 2PH and MAC.
    - For the ESA-Best results, ESA outperformed 2PH by 78.033%, 80.934%, and 41.853% in *low*, *moderate*, and *high* inventory cases, respectively. For the ESA-Ave results, ESA outperformed 2PH by 75.124%, 77.683%, and 4.941% in the respective cases.
    - Against MAC, ESA outperformed with ESA-Best results by 10.185%, 10.478%, and 4.990% for *low*, *moderate*, and *high* inventory cases, respectively. For the ESA-Ave results, ESA outperformed MAC by 7.275%, 7.218%, and 4.077% for each corresponding scenario.
3. Stability and Robustness Analysis: To assess the stability and robustness of ESA, we calculated the percentage deviation between the average fitness values (ESA-Ave)

and the best solution found (ESA-Best). High-inventory cases exhibited the smallest deviation at 0.913%, indicating strong stability. Deviations for *low* and *moderate* cases are 2.909% and 3.275%, respectively, reflecting consistent performance across inventory levels. Figure 2a–c illustrate the percentage gaps of ESA compared to MAC across instances per scenario, showing that ESA consistently maintains greater stability around the average gaps. Additionally, the gaps decrease as the scenarios become more challenging, demonstrating ESA's robustness.

4. Average Computational Time: ESA outperformed MAC in 68.33% of the sixty cases. Specifically, ESA demonstrated shorter computation times in 50% of the *low* inventory cases, 70% of the *moderate* inventory cases, and 85% of the *high* inventory cases. As instance difficulty increases ESA's performance becomes better. Figure 3 compares the computational times of MAC and ESA across three cases. While MAC's computational time increased significantly with harder instances, ESA maintained a stable trend, demonstrating consistent efficiency and highlighting its scalability and effectiveness.

In summary, ESA significantly outperformed 2PH and MAC, establishing new BKS for fifty-nine out of sixty cases. The low percentage deviations in solution quality highlight its robustness and stability, particularly in *high* inventory cases. Moreover, the ESA algorithm not only delivers superior solutions but also excels in computational efficiency, especially for harder cases.

Table 7. (**a**) SA vs. 2PH and MAC algorithms in large instances in *low*-inventory scenarios. (**b**) ESA vs. 2PH and MAC in large instances in moderate-inventory scenarios. (**c**) SA vs. 2PH and MAC in large instances in high-inventory scenarios. (**d**) Summary of ESA performance w.r.t. 2PH and MAC.

(a)

		Abdulkader et al. [3]					Efficient Simulated Annealing (ESA)				
Ins	BKS	2PH	MAC	Time (sec.)	%Gap (2PH)	%Gap (MAC)	ESA (Best)	ESA (Ave.)	Time (sec.)	%Gap* (Best)	%Gap* (Ave.)
b1	965.658	1631.63	1002.47	7.0	68.966%	3.812%	965.658	972.471	14.8	0.000%	0.706%
b2	1131.951	2057.50	1191.97	47.0	81.766%	5.302%	1131.951	1175.557	44.8	0.000%	3.852%
b3	1523.503	3006.19	1815.42	79.0	97.321%	19.161%	1523.503	1600.928	102.9	0.000%	5.082%
b4	1486.468	2830.16	1529.04	286.0	90.395%	2.864%	1486.468	1582.114	145.6	0.000%	6.434%
b5	1757.472	3478.72	1905.19	576.0	97.939%	8.405%	1757.472	1800.644	339.8	0.000%	2.456%
b6	1312.778	1774.35	1313.69	7.0	35.160%	0.069%	1312.778	1345.565	20.5	0.000%	2.498%
b7	1430.900	2461.83	1522.29	44.0	72.048%	6.387%	1430.900	1446.444	62.2	0.000%	1.086%
b8	1669.546	3545.11	2101.77	131.0	112.340%	25.889%	1669.546	1726.145	119.4	0.000%	3.390%
b9	2038.427	3528.98	2329.53	209.0	73.123%	14.281%	2038.427	2118.569	196.1	0.000%	3.932%
b10	2314.624	4916.75	3012.18	430.0	112.421%	30.137%	2314.624	2413.788	384.2	0.000%	4.284%
b11	1574.268	2432.56	1611.34	11.0	54.520%	2.355%	1574.268	1592.868	29.4	0.000%	1.182%
b12	1720.667	2695.34	1800.87	50.0	56.645%	4.661%	1720.667	1773.679	73.9	0.000%	3.081%
b13	2406.040	3936.67	2406.04	127.0	92.430%	17.611%	2045.768	2078.285	140.6	0.000%	1.589%
b14	2227.445	3826.14	2483.81	327.0	71.773%	11.509%	2227.445	2292.73	214.5	0.000%	2.931%
b15	2432.789	4496.05	2679.15	708.0	84.811%	10.127%	2432.789	2498.48	443.2	0.000%	2.700%
b16	1630.361	2254.87	1669.56	13.0	38.305%	2.404%	1630.361	1648.857	33.2	0.000%	1.134%
b17	1904.518	3020.80	1965.56	46.0	58.612%	3.205%	1904.518	1928.351	109.1	0.000%	1.251%
b18	2279.020	3963.52	2449.76	136.0	73.913%	7.492%	2279.020	2354.603	138.3	0.000%	3.316%
b19	2412.603	4933.86	2788.48	257.0	104.504%	15.580%	2412.603	2501.918	233.5	0.000%	3.702%
b20	2570.524	4721.26	2890.29	712.0	83.669%	12.440%	2570.524	2662.451	474.0	0.000%	3.576%
Ave.				210.1	78.033%	10.185%			166.0	0.000%	2.909%

(b)

		Abdulkader et al. [3]					Efficient Simulated Annealing (ESA)				
Ins	BKS	2PH	MAC	Time	%Gap	%Gap	ESA	ESA	Time (sec.)	%Gap *	%Gap *

				(sec.)	(2PH)	(MAC)	(Best)	(Ave.)		(Best)	(Ave.)
b21	830.576	1571.63	879.16	10.0	89.222%	5.849%	830.576	838.639	15.0	0.000%	0.971%
b22	985.593	1920.59	1083.66	85.0	94.866%	9.950%	985.593	1007.615	49.2	0.000%	2.234%
b23	1307.761	2699.23	1591.52	167.0	106.401%	21.698%	1307.761	1356.962	111.7	0.000%	3.762%
b24	1180.631	2305.09	1437.68	528.0	95.242%	21.772%	1180.631	1267.620	138.9	0.000%	7.368%
b25	1341.064	2700.41	1520.51	1836.0	101.363%	13.381%	1341.064	1421.077	336.2	0.000%	5.966%
b26	1177.616	1665.15	1180.83	11.0	41.400%	0.273%	1177.616	1182.247	21.2	0.000%	0.393%
b27	1326.928	2320.66	1329.26	73.0	74.890%	0.176%	1326.928	1348.179	62.9	0.000%	1.602%
b28	1504.426	3016.45	1692.41	279.0	100.505%	12.495%	1504.426	1586.755	129.2	0.000%	5.472%
b29	1685.024	3302.39	2016.40	567.0	95.985%	19.666%	1685.024	1779.461	193.3	0.000%	5.604%
b30	1812.834	3918.97	2399.60	1407.0	116.179%	32.367%	1812.834	1970.385	435.6	0.000%	8.691%
b31	1495.790	1993.49	1495.79	16.0	33.273%	0.000%	1500.196	1514.549	29.9	0.295%	1.254%
b32	1583.288	2713.01	1656.86	76.0	71.353%	4.647%	1583.288	1604.756	80.6	0.000%	1.356%
b33	1715.007	3393.34	1799.64	262.0	97.862%	4.935%	1715.007	1772.595	147.7	0.000%	3.358%
b34	1821.977	3127.49	2018.50	740.0	71.654%	10.786%	1821.977	1883.839	230.5	0.000%	3.395%
b35	2008.901	3742.21	2290.95	2141.0	86.281%	14.040%	2008.901	2092.139	458.0	0.000%	4.143%
b36	1496.488	2032.06	1550.01	15.0	35.789%	3.577%	1496.488	1518.099	37.3	0.000%	1.444%
b37	1833.898	3130.52	1939.50	73.0	70.703%	5.758%	1833.898	1880.449	104.1	0.000%	2.538%
b38	1944.259	3433.23	2088.58	283.0	76.583%	7.423%	1944.259	1964.290	162.2	0.000%	1.030%
b39	1996.767	3824.46	2244.14	656.0	91.533%	12.389%	1996.767	2023.750	280.6	0.000%	1.351%
b40	2051.460	3447.89	2229.43	2077.0	68.070%	8.675%	2051.460	2124.600	480.9	0.000%	3.565%
Ave.				565.1	80.958%	10.493%			175.3	0.015%	3.275%

(c)

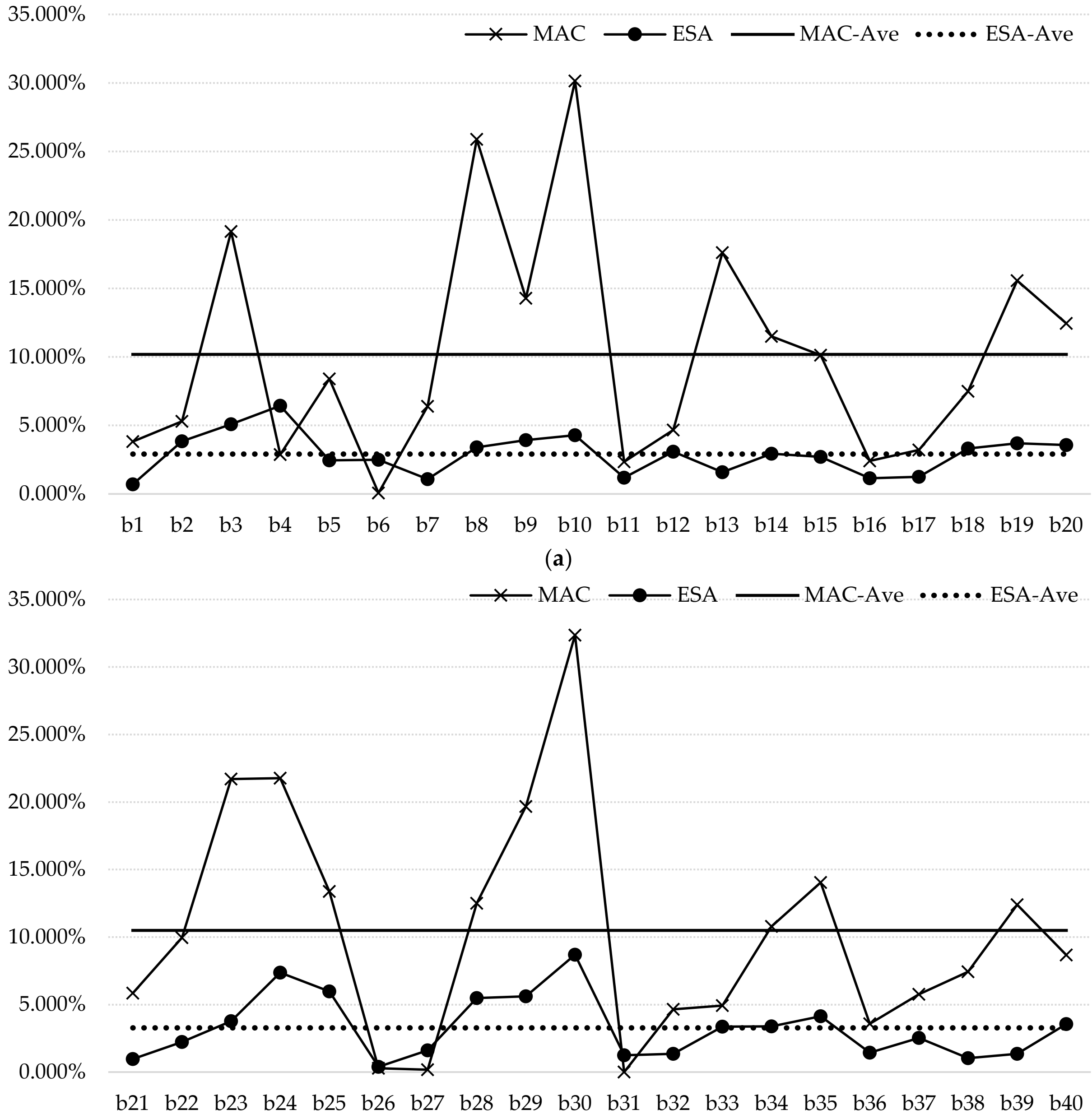
					Abdulkader et al. [3]		Efficient Simulated Annealing (ESA)				
Ins	**BKS**	**2PH**	**MAC**	**Time (sec.)**	**%Gap (2PH)**	**%Gap (MAC)**	**ESA (Best)**	**ESA (Ave.)**	**Time (sec.)**	**%Gap (Best)**	**%Gap (Ave.)**
b41	692.317	897.55	711.29	16.0	29.644%	2.741%	692.317	692.317	14.7	0.000%	0.000%
b42	853.859	1287.81	875.24	143.0	50.822%	2.504%	853.859	855.624	49.2	0.000%	0.207%
b43	1047.763	1531.06	1132.05	358.0	46.127%	8.044%	1047.763	1054.713	108.8	0.000%	0.663%
b44	1053.330	1636.54	1224.11	978.0	55.368%	16.213%	1053.330	1079.247	148.9	0.000%	2.460%
b45	1155.630	1551.75	1273.86	2085.0	34.277%	10.231%	1155.630	1176.100	332.5	0.000%	1.771%
b46	988.557	1264.32	996.93	22.0	27.896%	0.847%	988.557	988.557	26.1	0.000%	0.000%
b47	1027.516	1488.07	1080.34	159.0	44.822%	5.141%	1027.516	1036.107	66.3	0.000%	0.836%
b48	1188.503	1815.22	1252.35	559.0	52.732%	5.372%	1188.503	1216.658	121.3	0.000%	2.369%
b49	1471.915	2242.40	1594.02	1167.0	52.346%	8.296%	1471.915	1504.279	203.3	0.000%	2.199%
b50	1503.470	2459.52	1691.42	4126.0	63.590%	12.501%	1503.470	1521.713	439.8	0.000%	1.213%
b51	1302.884	1660.91	1302.89	33.0	27.479%	0.000%	1302.884	1303.162	33.7	0.000%	0.021%
b52	1272.812	1740.66	1300.98	156.0	36.757%	2.213%	1272.812	1278.270	80.2	0.000%	0.429%
b53	1391.373	2096.76	1421.77	605.0	50.697%	2.185%	1391.373	1393.505	139	0.000%	0.153%
b54	1601.141	2226.39	1640.60	1370.0	39.050%	2.464%	1601.141	1608.418	263.5	0.000%	0.454%
b55	1638.572	2518.16	1763.31	5321.0	53.680%	7.613%	1638.572	1666.076	484.8	0.000%	1.679%
b56	1311.628	1550.71	1311.63	36.0	18.228%	0.000%	1311.628	1311.628	39.9	0.000%	0.000%
b57	1423.865	1835.39	1468.12	203.0	28.902%	3.108%	1423.865	1436.874	113.7	0.000%	0.914%
b58	1601.873	2276.94	1654.92	791.0	42.142%	3.312%	1601.873	1604.410	166.5	0.000%	0.158%
b59	1532.106	2061.86	1575.67	1262.0	34.577%	2.843%	1532.106	1546.861	277.8	0.000%	0.963%
b60	1587.046	2347.76	1653.30	4549.0	47.933%	4.175%	1587.046	1615.076	478.5	0.000%	1.766%
	Ave.			1196.9	41.853%	4.990%			179.4	0.000%	0.913%

(d)

	Mean %Gaps w.r.t. BKS per Scenario	ESA Outperforms 2PH by:	ESA Outperforms MAC by:

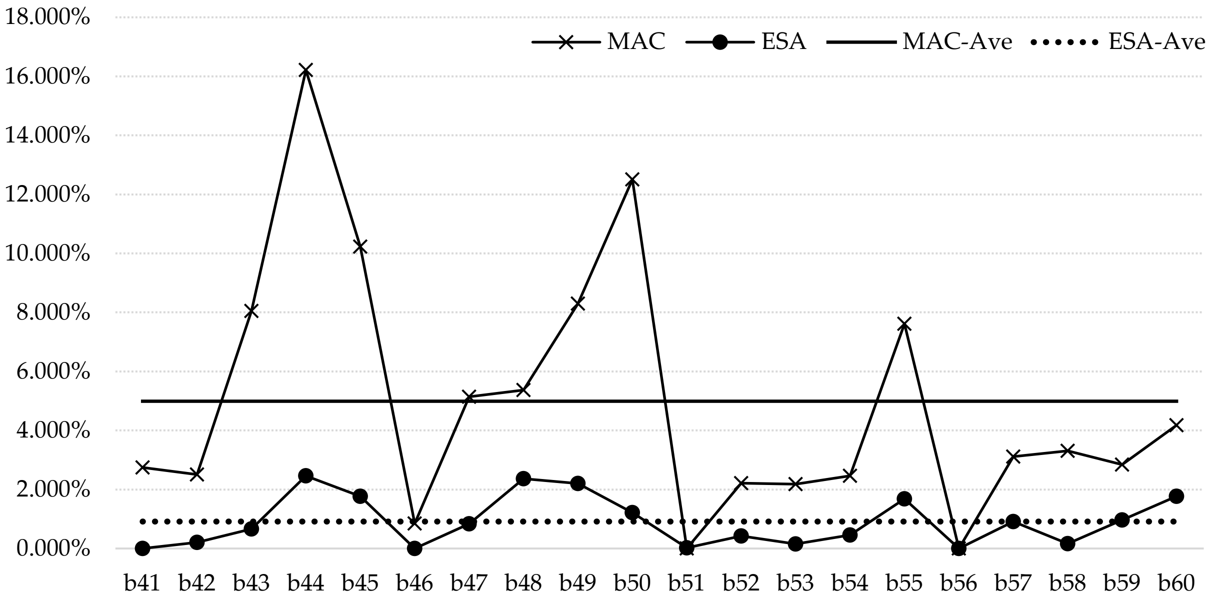
Case	ESA (Best)	ESA (Ave)	2PH	MAC	ESA (Best)	ESA (Ave)	ESA (Best)	ESA (Ave)
Low	0.000%	2.909%	78.033%	10.185%	−78.033%	−75.124%	−10.185%	−7.275%
Moderate	0.015%	3.275%	80.958%	10.493%	−80.943%	−77.683%	−10.478%	−7.218%
High	0.000%	0.913%	41.853%	4.990%	−41.853%	−40.941%	−4.990%	−4.077%

* $\%Gap = \frac{obj_{Heuristic} - obj_{BKS}}{obj_{BKS}} \times 100.$



(**a**)

(**b**)



(**c**)

Figure 2. Average %Gap with respect to BKS across inventory cases: MAC vs. ESA. (**a**) *Low*; (**b**) *Moderate*; and (**c**) *High*.

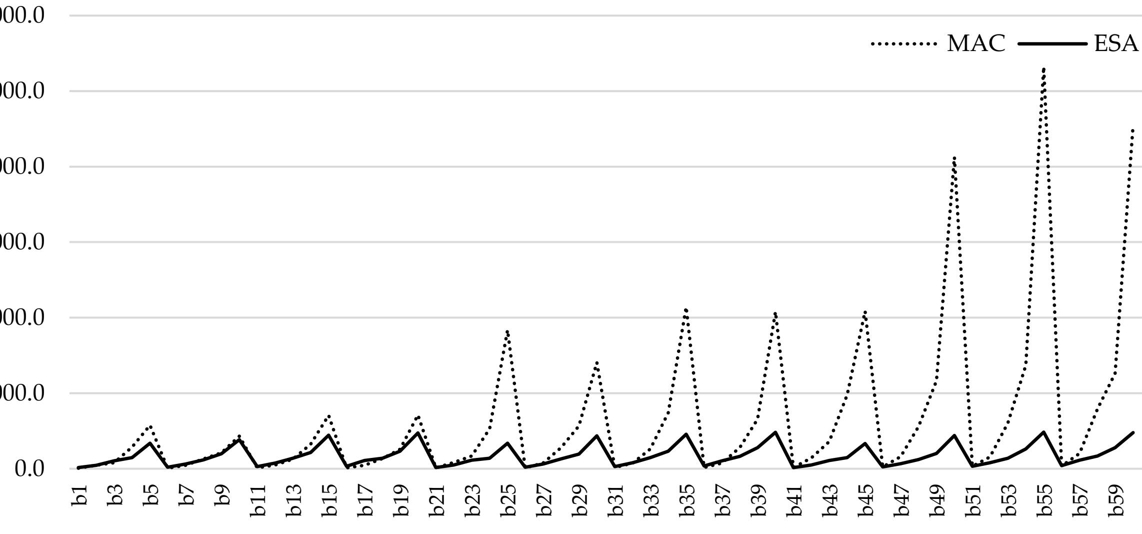


Figure 3. CPU time (sec) across 60 cases of MAC vs. ESA.

5. Summary and Conclusions

The VRPO is a hybrid VRP with PDP and SPDP features. As omnichannel operations evolve, last-mile distribution networks are becoming increasingly complex, making the pickup and delivery problems relevant than ever. To solve the VRPO, we tailor an efficient SA-based algorithm with two efficiency features: the non-improving solution parameter (N_{nimp}) and the unit penalty function (*UnitP*) for constraint handling. The experimental study demonstrated ESA's performance in terms of solution quality, average quality, robustness, and efficiency.

Building on the study of Abdulkader et al. [3], the experiment revealed the following results: (i) ESA obtained optimal solutions for all twenty small instances, (ii) ESA established fifty-nine new BKSs out of sixty large instances, (iii) ESA outperformed the MAC algorithm by 6.190% in terms of solution quality, (iv) ESA was faster than MAC in 68.3%

of the sixty cases, and (v) ESA demonstrated a better robustness than MAC, as evidenced by the stable and consistent trend of computational times across all sixty instances.

The PDP domains in the omnichannel context offer a promising opportunity to test and demonstrate various solution approaches, including exact methods, heuristics, metaheuristics, and hybrids. This study addresses the gap regarding the underexplored use of SA-based algorithms for PDPs in pre- or post-omnichannel settings. As a flexible framework, it is recommended for further exploration of other SA components. Testing VRPO instances with neighborhood-based metaheuristics like variable neighborhood search (VNS), large neighborhood search (LNS), and variable neighborhood descent (VND), which have gained traction recently, would be valuable. Additionally, applying the framework to new omnichannel distribution network variants—such as managing customer returns, delivering goods to secondary markets or recycling, and warehouse milk-run deliveries—offers further opportunities to extend the VRPO.

The landscape of metaheuristics for solving vehicle routing problems has significantly evolved, with recent advancements particularly emphasizing hybrid and AI-driven approaches. For instance, a hybrid simulated annealing and variable neighborhood search algorithm was developed for a novel variant of electric vehicle routing (EVRP) called close–open EVRP (COEVRP) [36]. The use of such hybrid approaches has been instrumental in achieving high-quality solutions, especially in complex VRP variants. Furthermore, the integration of adaptive algorithms [37] and reinforcement learning [30] into the simulated annealing algorithm offers a better position for it to advance further.

Author Contributions: Conceptualization, V.F.Y. and C.-H.L.; Methodology, V.F.Y., R.S.M. and S.-W.L.; Validation, C.-H.L., R.S.M. and S.-W.L.; Formal analysis, C.-H.L., S.-W.L. and K.-F.C.; Investigation, V.F.Y. and S.-W.L.; Data curation, C.-H.L., S.-W.L. and K.-F.C.; Writing—original draft, C.-H.L. and R.S.M.; Writing—review & editing, V.F.Y., S.-W.L. and K.-F.C.; Visualization, C.-H.L. and R.S.M.; Supervision, V.F.Y. and S.-W.L.; Project administration, V.F.Y.; Funding acquisition, V.F.Y. and S.-W.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the National Science and Technology Council, the Republic of China (Taiwan) under Grant MOST111-2410-H-011-020-MY3 for the first author, and was partially supported by the National Science and Technology Council, Republic of China (Taiwan) under Grant NSTC112-2410-H-182-002-MY3 for the corresponding author.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lokin, F.C.J. Procedures for Travelling Salesman Problems with Additional Constraints. *Eur. J. Oper. Res.* **1979**, *3*, 135–141. https://doi.org/10.1016/0377-2217(79)90099-7.
2. Berbeglia, G.; Cordeau, J.-F.; Gribkovskaia, I.; Laporte, G. Static Pickup and Delivery Problems: A Classification Scheme and Survey. *TOP* **2007**, *15*, 1–31. https://doi.org/10.1007/S11750-007-0009-0.
3. Abdulkader, M.M.S.; Gajpal, Y.; ElMekkawy, T.Y. Vehicle Routing Problem in Omni-Channel Retailing Distribution Systems. *Int. J. Prod. Econ.* **2018**, *196*, 43–55. https://doi.org/10.1016/J.IJPE.2017.11.011.
4. Braekers, K.; Caris, A.; Janssens, G.K. Exact and Meta-Heuristic Approach for a General Heterogeneous Dial-a-Ride Problem with Multiple Depots. *Transp. Res. Part B Methodol.* **2014**, *67*, 166–186. https://doi.org/10.1016/J.TRB.2014.05.007.
5. Parragh, S.N.; Doerner, K.F.; Hartl, R.F. A Survey on Pickup and Delivery Problems. Part I: Transportation Between Customers and Depot. *J. Für Betriebswirtschaft* **2008**, *58*, 21–51. https://doi.org/10.1007/s11301-008-0033-7.
6. Battarra, M.; Cordeau, J.-F.; Iori, M. Chapter 6: Pickup-and-Delivery Problems for Goods Transportation. In *Vehicle Routing: Problems, Methods, and Applications*; MOS-SIAM Series on Optimization; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2014; pp. 161–191.
7. Parragh, S.N.; Doerner, K.F.; Hartl, R.F. A Survey on Pickup and Delivery Problems: Part II: Transportation Between Pickup and Delivery Locations. *J. Für Betriebswirtschaft* **2008**, *58*, 81–117. https://doi.org/10.1007/S11301-008-0036-4.
8. Beck, N.; Rygl, D. Categorization of Multiple Channel Retailing in Multi-, Cross-, and Omni-Channel Retailing for Retailers and Retailing. *J. Retail. Consum. Serv.* **2015**, *27*, 170–178. https://doi.org/10.1016/j.jretconser.2015.08.001.

9. Martins, L.D.C.; Bayliss, C.; Juan, A.A.; Panadero, J.; Marmol, M. A Savings-Based Heuristic for Solving the Omnichannel Vehicle Routing Problem with Pick-up and Delivery. *Transp. Res. Procedia* **2020**, *47*, 83–90. https://doi.org/10.1016/J.TRPRO.2020.03.082.
10. Martins, L.D.C.; Bayliss, C.; Copado-Méndez, P.J.; Panadero, J.; Juan, A.A. A Simheuristic Algorithm for Solving the Stochastic Omnichannel Vehicle Routing Problem with Pick-up and Delivery. *Algorithms* **2020**, *13*, 237. https://doi.org/10.3390/a13090237.
11. Bayliss, C.; Martins, L.d.C.; Juan, A.A. A Two-Phase Local Search with a Discrete-Event Heuristic for the Omnichannel Vehicle Routing Problem. *Comput. Ind. Eng.* **2020**, *148*, 106695. https://doi.org/10.1016/j.cie.2020.106695.
12. Sawicki, P.; Sawicka, H. Optimisation of the Two-Tier Distribution System in Omni-Channel Environment. *Energies* **2021**, *14*, 7700. https://doi.org/10.3390/en14227700.
13. Janjevic, M.; Merchán, D.; Winkenbach, M. Designing Multi-Tier, Multi-Service-Level, and Multi-Modal Last-Mile Distribution Networks for Omni-Channel Operations. *Eur. J. Oper. Res.* **2021**, *294*, 1059–1077. https://doi.org/10.1016/j.ejor.2020.08.043.
14. Guerrero-Lorente, J.; Gabor, A.F.; Ponce-Cueto, E. Omnichannel Logistics Network Design with Integrated Customer Preference for Deliveries and Returns. *Comput. Ind. Eng.* **2020**, *144*, 106433. https://doi.org/10.1016/j.cie.2020.106433.
15. Liu, P.; Hendalianpour, A.; Feylizadeh, M.; Pedrycz, W. Mathematical Modeling of Vehicle Routing Problem in Omni-Channel Retailing. *Appl. Soft Comput.* **2022**, *131*, 109791. https://doi.org/10.1016/J.ASOC.2022.109791.
16. Hendalianpour, A.; Fakhrabadi, M.; Sangari, M.S.; Razmi, J. A Combined Benders Decomposition and Lagrangian Relaxation Algorithm for Optimizing a Multi-Product, Multi-Level Omni-Channel Distribution System. *Sci. Iran.* **2022**, *29*, 355–371. https://doi.org/10.24200/sci.2020.53644.3349.
17. Li, Y. A Selective Many-to-Many Pickup and Delivery Problem with Handling Cost in the Omni-Channel Last-Mile Delivery. *IEEE Access* **2022**, *10*, 111284–111296. https://doi.org/10.1109/ACCESS.2022.3215700.
18. Yang, J.; Li, Y. A Multicommodity Pickup and Delivery Problem with Time Windows and Handling Time in the Omni-Channel Last-Mile Delivery. *Int. Trans. Oper. Res.* **2023**. https://doi.org/10.1111/itor.13362.
19. Qiu, R.; Yuan, M.; Sun, M.; Fan, Z.P.; Xu, H. Optimizing Omnichannel Retailer Inventory Replenishment Using Vehicle Capacity-Sharing with Demand Uncertainties and Service Level Requirements. *Eur. J. Oper. Res.* **2025**, *320*, 417–432. https://doi.org/10.1016/J.EJOR.2024.08.005.
20. Li, N.; Wang, Z. Vehicle Routing Problem for Omnichannel Retailing Including Multiple Types of Time Windows and Products. *Comput. Oper. Res.* **2025**, *173*, 106828. https://doi.org/10.1016/j.cor.2024.106828.
21. Wang, C.; Mu, D.; Zhao, F.; Sutherland, J.W. A Parallel Simulated Annealing Method for the Vehicle Routing Problem with Simultaneous Pickup-Delivery and Time Windows. *Comput. Ind. Eng.* **2015**, *83*, 111–122. https://doi.org/10.1016/j.cie.2015.02.005.
22. Avci, M.; Topaloglu, S. An Adaptive Local Search Algorithm for Vehicle Routing Problem with Simultaneous and Mixed Pickups and Deliveries. *Comput. Ind. Eng.* **2015**, *83*, 15–29. https://doi.org/10.1016/J.CIE.2015.02.002.
23. Danloup, N.; Allaoui, H.; Goncalves, G. A Comparison of Two Meta-Heuristics for the Pickup and Delivery Problem with Transshipment. *Comput. Oper. Res.* **2018**, *100*, 155–171. https://doi.org/10.1016/J.COR.2018.07.013.
24. Koç, Ç.; Laporte, G.; Tükenmez, İ. A Review of Vehicle Routing with Simultaneous Pickup and Delivery. *Comput. Oper. Res.* **2020**, *122*, 104987. https://doi.org/10.1016/J.COR.2020.104987.
25. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by Simulated Annealing. *Science* **1983**, *220*, 671–680. https://doi.org/10.1126/science.220.4598.671.
26. İlhan, İ. An Improved Simulated Annealing Algorithm with Crossover Operator for Capacitated Vehicle Routing Problem. *Swarm Evol. Comput.* **2021**, *64*, 100911. https://doi.org/10.1016/j.swevo.2021.100911.
27. Lin, S.W.; Cheng, C.Y.; Pourhejazy, P.; Ying, K.C. Multi-Temperature Simulated Annealing for Optimizing Mixed-Blocking Permutation Flowshop Scheduling Problems. *Expert Syst. Appl.* **2021**, *165*, 113837. https://doi.org/10.1016/j.eswa.2020.113837.
28. Yu, V.F.; Anh, P.T.; Gunawan, A.; Han, H. A Simulated Annealing with Variable Neighborhood Descent Approach for the Heterogeneous Fleet Vehicle Routing Problem with Multiple Forward/Reverse Cross-Docks. *Expert Syst. Appl.* **2024**, *237*, 121631. https://doi.org/10.1016/j.eswa.2023.121631.
29. Leite, N.; Melício, F.; Rosa, A.C. A Fast Simulated Annealing Algorithm for the Examination Timetabling Problem. *Expert Syst. Appl.* **2019**, *122*, 137–151. https://doi.org/10.1016/j.eswa.2018.12.048.
30. Rodríguez-Esparza, E.; Masegosa, A.D.; Oliva, D.; Onieva, E. A New Hyper-Heuristic Based on Adaptive Simulated Annealing and Reinforcement Learning for the Capacitated Electric Vehicle Routing Problem. *Expert Syst. Appl.* **2024**, *252*, 124197. https://doi.org/10.1016/j.eswa.2024.124197.
31. Yu, V.F.; Indrakarna, P.A.Y.; Ngurah, A.A.; Redi, P.; Lin, S.-W. Simulated Annealing with Mutation Strategy for the Share-a-Ride Problem with Flexible Compartments. *Mathematics* **2021**, *9*, 2320. https://doi.org/10.3390/math9182320.
32. Lin, S.W.; Yu, V.F. Solving the Team Orienteering Problem with Time Windows and Mandatory Visits by Multi-Start Simulated Annealing. *Comput. Ind. Eng.* **2017**, *114*, 195–205. https://doi.org/10.1016/J.CIE.2017.10.020.
33. Ying, K.C.; Lin, S.W. Minimizing Total Completion Time in the No-Wait Jobshop Scheduling Problem Using a Backtracking Metaheuristic. *Comput. Ind. Eng.* **2022**, *169*, 108238. https://doi.org/10.1016/J.CIE.2022.108238.
34. Lin, S.W.; Cheng, C.Y.; Pourhejazy, P.; Ying, K.C.; Lee, C.H. New Benchmark Algorithm for Hybrid Flowshop Scheduling with Identical Machines. *Expert Syst. Appl.* **2021**, *183*, 115422. https://doi.org/10.1016/J.ESWA.2021.115422.
35. Yu, V.F.; Susanto, H.; Yeh, Y.-H.; Lin, S.-W.; Huang, Y.-T. The Vehicle Routing Problem with Simultaneous Pickup and Delivery and Parcel Lockers. *Mathematics* **2022**, *10*, 920. https://doi.org/10.3390/math10060920.

36. Stamadianos, T.; Kyriakakis, N.A.; Marinaki, M.; Marinakis, Y. A Hybrid Simulated Annealing and Variable Neighborhood Search Algorithm for the Close-Open Electric Vehicle Routing Problem. *Ann. Math. Artif. Intell.* **2023**. https://doi.org/10.1007/s10472-023-09858-x.
37. Xiao, S.; Peng, P.; Zheng, P.; Wu, Z. A Hybrid Adaptive Simulated Annealing and Tempering Algorithm for Solving the Half-Open Multi-Depot Vehicle Routing Problem. *Mathematics* **2024**, *12*, 947. https://doi.org/10.3390/math12070947.