algorithms

Article

AASA: A Priori Adaptive Splitting Algorithm for the Split Delivery Vehicle Routing Problem

Nariman Torkzaban [1,*,†], Anousheh Gholami [1,†], John S. Baras [1] and Bruce L. Golden [2,*]

[1] Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742, USA; anousheh@umd.edu (A.G.); baras@umd.edu (J.S.B.)
[2] Robert H. Smith School of Business, University of Maryland, College Park 20742, MD, USA
* Correspondence: narimant@umd.edu (N.T.); bgolden@umd.edu (B.L.G.)
† These authors contributed equally to this work.

Abstract: The split delivery vehicle routing problem (SDVRP) is a relaxed variant of the capacitated vehicle routing problem (CVRP) where the restriction that each customer is visited precisely once is removed. Compared with CVRP, the SDVRP allows a reduction in the total cost of the routes traveled by vehicles. The exact methods to solve the SDVRP are computationally expensive. Moreover, the complexity and difficult implementation of the state-of-the-art heuristic approaches hinder their application in real-life scenarios of the SDVRP. In this paper, we propose an easily understandable and effective approach to solve the SDVPR based on an a priori adaptive splitting algorithm (AASA) that improves the existing state of the art on a priori split strategy in terms of both solution accuracy and time complexity. In this approach, the demand of the customers is split into smaller demand values using a splitting rule in advance. Consequently, the original SDVRP instance is converted to a CVRP instance which is solved using an existing CVRP solver. While the proposed a priori splitting rule in the literature is fixed for all customers regardless of their demand and location, we suggest an adaptive splitting rule that takes into account the distance of the customers to the depot and their demand values. Our experiments show that AASA can generate solutions comparable to the state of the art, but much faster.

Keywords: split delivery vehicle routing problem; capacitated vehicle routing problem; splitting rule

Citation: Torkzaban, N.; Gholami, A.; Baras, J.S.; Golden, B.L. AASA: A Priori Adaptive Splitting Algorithm for the Split Delivery Vehicle Routing Problem. *Algorithms* **2024**, *17*, 396. https://doi.org/10.3390/a17090396

Academic Editor: Frank Werner

Received: 6 June 2024
Revised: 17 August 2024
Accepted: 29 August 2024
Published: 6 September 2024

1. Introduction

The split delivery vehicle routing problem (SDVRP) introduced by [1] is a relaxation of the traditional capacitated vehicle routing problem (CVRP), where each customer can be visited more than once. In both CVRP and SDVRP, a number of identical vehicles having limited capacity serve a set of customers with given demands. The vehicles depart a depot and return to the depot after visiting customers. The pairwise travel costs between customers and between customers and the depot are given. The objective is to minimize the total travel cost of the vehicles. The difference between CVRP and SDVRP is that in contrast to CVRP, where each customer is required to be visited once and by only one vehicle, SDVRP allows multiple visits to a customer. Therefore, the demand of a customer may be split among multiple vehicles. The authors in [2] showed that splitting the customer demands potentially reduces the total cost by up to 50%. Furthermore, they show that there always exists an optimal solution to the SDVRP in which there is no k-split cycle and no two routes have more than one common customer.

Apart from environmental benefits, such as reduced emissions and energy efficiency, by allowing split deliveries, SDVRP can reduce the total distance traveled and the number of vehicles required. This optimization leads to lower fuel consumption, reduced vehicle wear and tear, and decreased maintenance costs. For businesses with large delivery fleets, these savings can be substantial. Split delivery may reduce labor, inventory, and storage costs as efficient routing reduces the time drivers spend on the road, which translates to lower labor costs. Reduced travel time also minimizes overtime and improves overall productivity. Moreover, split deliveries enable better alignment with demand patterns. By avoiding the need to stockpile large quantities of inventory, businesses can reduce warehousing costs and minimize the risk of overstocking or obsolescence. These findings justify the essence of formulating and solving the SDVRP.

State-of-the-art research has applied the SDVRP to various practical scenarios. For instance, Mullaseril et al. [3] addressed the problem of distributing feed to cattle on a large ranch in Arizona, where inaccuracies in feed loading necessitate split deliveries. Their approach, modeled as a capacitated rural postman problem with time windows, demonstrated that allowing split deliveries significantly reduced the total distance traveled by the fleet in most cases. Sierksma and Tijssen [4] studied the crew exchange process between offshore gas platforms in the North Sea. They formulated this as a discrete split delivery routing problem and employed integer programming with column generation and a cluster-and-route procedure. Their experiments showed that while their column generation approach (CGRP) was more accurate, the cluster-and-route (CR) method offered faster computation times. Song et al. [5] explored the distribution of newspapers from printing plants to agents in South Korea. They implemented a two-phase solution procedure, including agent allocation and split delivery scheduling, which improved the overall delivery efficiency and reduced costs by 15% compared to manual methods. More recently, within the context of waste management, ref. [6] utilized the SDVRP framework to model and solve complex waste collection and transportation challenges, incorporating realistic collection policies and offering insights into optimizing operational costs and fleet management, where they showed how an appropriate collection policy can significantly reduce fleet operating costs by 40–60% and fleet size by approximately 50% without causing overflow.

The SDVRP is known to be an NP-hard problem [7]. Therefore, exact solutions are only applicable to the SDVRP instances of smaller scale. Such methods typically formulate the SDVRP as a mixed integer linear program (MILP) [8] and propose different variants of branching algorithms such as branch-and-cut [9,10], branch-and-price [11,12], and branch-and-price-and-cut [13,14] for various versions of the SDVRP. However, with increasing the size of the problem, the corresponding MILP models will soon become intractable for exact solutions. To tackle this issue, various traditional approximation techniques, heuristics and metaheuristics have been proposed to deal with the high time-complexity of the large SDVRP models. For instance, a max–min clustering before solving the optimization problem is proposed in [15]. Tabu search has been used in several papers [16–18]. In [16], the authors proposed an approach relying on simulated annealing to solve the SDVRP. Other lines of research propose methods utilizing ant colony optimization [19,20], genetic algorithms [21,22], and particle swarm optimization [23,24], to solve the SDVRP. Using column generation is another optimization technique that has been applied to the SDVRP by [8,25]. The authors in [26] applied the cutting plane method to tackle the time complexity of the SDVRP.

Recently, the authors in [27] introduced a metaheuristic approach incorporating several mathematical programming components within an iterative local search framework. We refer to this method as ILS-MIP in the rest of the paper. ILS-MIP starts from an initial set of solutions obtained using the I1 Solomon heuristic followed by a perturbation step (to escape local optima), a local search step using classical neighborhood search heuristics, and several steps based on hybrid components. The hybrid components are called only if the incumbent solution of the search is not improved after a number of steps. The hybrid components encompass a mixed integer program (MIP)-based improvement heuristic performing two operations of inserting splits or removing potential unnecessary splits. Another used hybrid component is based on converting the current SDVRP solution to a CVRP instance and using a hybrid genetic search (HGS) framework [28] specialized to solve a CVRP. An MIP model is also used in the third component of the hybrid step where a residual problem is constructed by removing the edges that are not used in any solution of the current set of solutions. The residual problem is then solved using an MIP solver. Finally, the fourth component uses the branch-and-cut (BC) framework proposed in [29]. While ILS-MIP is a powerful and effective methodology, its complexity and significant runtime for larger instances makes it challenging for addressing real-world problems.

Although the state-of-the-art algorithms for the SDVRP are known for their ability to produce high-quality solutions, they often pose challenges in terms of complexity and practical implementation. In response to these challenges, the authors in [30] introduced an efficient algorithm that decomposes the SDVRP into two sub-problems. First, a demand-splitting rule is applied to all customers. The idea of splitting the customer demands was introduced in [31]; the authors propose a heuristic solution for SDVRP with time windows based on the column generation approach. In this paper, it is assumed that the demand of each customer is served by a number of orders selected from a set of feasible orders where each order is a pre-determined discrete split of the customer demand. The authors in [30] proposed applying the a priori splitting strategy in which the demand of each customer is split into smaller demands, each representing a new customer located in the same position as the original customer. As a result, a new problem instance is generated with an increased number of customers. Second, the new problem is assumed to be an instance of the CVRP which is solved using existing powerful CVRP solvers. The transformation of the SDVRP to a CVRP not only facilitates the implementation of SDVRP but also enables leveraging the rich literature on the CVRP. The obtained solution for the resulting CVRP is then translated back to the original SDVRP.

Regarding the splitting strategy, the solutions proposed in both [30,31] are based on a static splitting rule without taking into account the specific characteristics of the problem, such as customers' location and demand. In the method proposed by [30], a fixed splitting rule that is inspired by the US coin denominations is applied to all customer demands and the VRPH solver introduced by [32] is used to solve the resulting CVRP instance. We refer to this algorithm as VRPHAS in the rest of this paper. The numerical evaluation of VRPHAS illustrates its ability to produce acceptable sub-optimal solutions in much less time compared to the state-of-the-art heuristic algorithms. However, the adoption of the introduced splitting rule is not well justified. Instead of applying a fixed rule to all customers, a splitting rule that is adaptive, based on the specific characteristics of a customer can potentially result in an improved solution for the SDVRP. In this paper, we study the problem of defining a good splitting rule for each customer. To this end, we introduce an a priori adaptive splitting algorithm (AASA). In combination with the VRPH solver, AASA results in an improved performance in terms of the optimality gap without significantly increasing the computational complexity of VRPHAS. AASA achieves this by taking into account the distance between the customers and the depot, the value of the customers' demand along with the vehicles' capacity. Similar to VRPHAS, AASA can be used with any CVRP solver. Thus, it provides a method to solve the SDVRP that is easily understandable, can be implemented simply, and generates high-quality solutions very quickly.

The rest of the paper is organized as follows. In Section 2.1, we provide a formal definition of the SDVRP. Our proposed solution is explained in Section 2.2. Section 3 presents the numerical results. Finally, the conclusion is discussed in Section 4.

2. Problem Definition and the Proposed Solution

2.1. SDVRP Definition

Let $G = (V, E)$ denote an undirected and weighted complete graph representing the network of customers and a depot. The vertex set $V = \{0, \ldots, n\}$ represents the depot denoted by the vertex 0 and n customers indexed as $1, \ldots, n$. A non-negative weight c_{ij} is associated with each edge of the graph $(i, j) \in E$ that stands for the cost of traveling between customers i and j. Moreover, each customer $i \in V$ has a positive demand, denoted by d_i. A set of M vehicles is available to serve the customers' demands. Each vehicle has a limited capacity denoted by Q. The objective of the SDVRP is to find a route for each vehicle such that the demands of all customers are satisfied with the minimum total cost. In this paper, we assume that the cost of traveling between customers $i, j \in V$ is their Euclidean distance denoted by the function *dist*, i.e., $c_{ij} = dist(i, j)$.

2.2. Proposed Heuristic Algorithm for SDVRP

In the following, we first describe the idea of the a priori splitting rule proposed by [30] and then discuss the core algorithmic idea of AASA based on two motivational observations. We then formalize AASA in detail.

A Priori Splitting Rule

The proposed splitting rule in [30] is inspired by the US coin denominations. The authors propose two splitting options 20/10/5/1, and 25/10/5/1. The first option replaces customer i, with the demand of d_i, by m^i_{20}, m^i_{10}, m^i_5 and m^i_1 customers with the demand values of $0.2Q$, $0.1Q$, $0.05Q$ and $0.01Q$, respectively. Hence, we have:

$$m^i_{20} = max\{m \in \mathbb{Z}^+ \cup \{0\} : 0.2Qm \le d_i\}; \tag{1}$$

$$m^i_{10} = max\{m \in \mathbb{Z}^+ \cup \{0\} : 0.1Qm \le d_i - 0.2Qm^i_{20}\}; \tag{2}$$

$$m^i_5 = max\{m \in \mathbb{Z}^+ \cup \{0\} : 0.05Qm \le d_i - 0.2Qm^i_{20} - 0.1Qm^i_{10}\}; \tag{3}$$

$$m^i_1 = max\{m \in \mathbb{Z}^+ \cup \{0\} : 0.01Qm \le d_i - 0.2Qm^i_{20} - 0.1Qm^i_{10} - 0.05Qm^i_5\}. \tag{4}$$

A similar breakdown can be given for the second splitting rule option. Using the splitting rule 20/10/5/1, the demand of customer i is split into $m^i = m^i_{20} + m^i_{10} + m^i_5 + m^i_1$ smaller values. The customer i is then replaced by m^i customers located at the same position as customer i and with the demand values resulting from (1)–(4). After applying the splitting rule to all customers, a new graph is constructed with $m = \sum_{i=1}^{n} m^i$ vertices (customers). The new graph is considered as an instance of the CVRP and the VRPH solver is used to solve it. The resulting solution is also a solution to the original SDVRP instance. In the next section, we justify the essence of the adaptation of the splitting rule and propose the AASA algorithm.

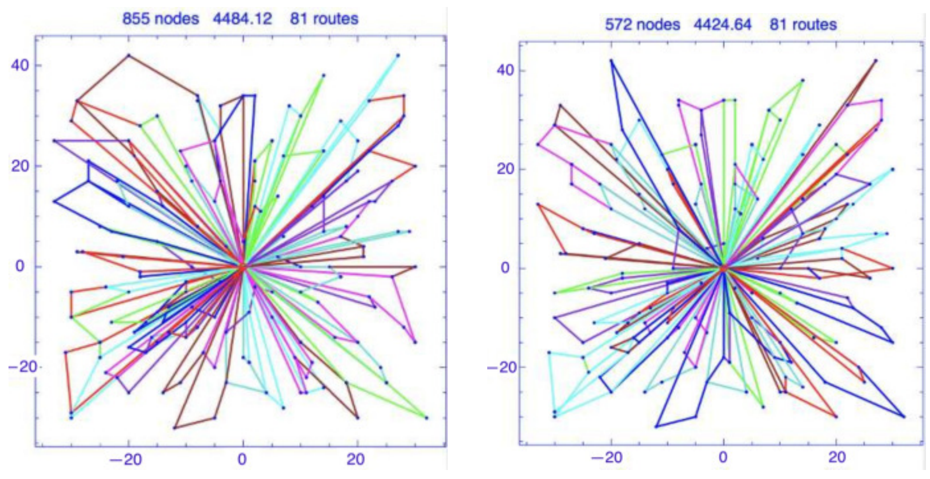
2.3. A Priori Adaptive Splitting Algorithm (AASA)

The a priori splitting rule described by Equations (1)–(4) is a fixed rule, i.e., a single splitting rule is applied to all customers regardless of their demand and location. However, the information regarding the customers' demand and location can be leveraged in the vehicle routing decisions towards reducing the total travel cost as well as the size of the resulting CVRP instance. We propose an adaptive splitting rule based on the location and demand information that is shown to effectively improve the solution. In the following, we first present the motivation behind the main building blocks of our proposed splitting rule. We then explain the AASA algorithm.

2.3.1. Motivation

In this section, we present the motivation behind our proposed splitting algorithm through illustrative examples. We study the impact of the splitting rule granularity on the total travel cost and the solution runtime considering customers' demand values and their distances from the depot. Our investigations reveal the benefits of using coarser/finer splitting rules for different customers depending on their demand and location. By a coarser splitting rule, we mean one that entails larger splitting portions as opposed to a finer splitting rule where the portions are smaller. For instance, for a demand point with $d = 200$, the splitting rule $200 = 128 + 64 + 8$ is coarser compared to the rule $200 = 3 \times 64 + 8$.

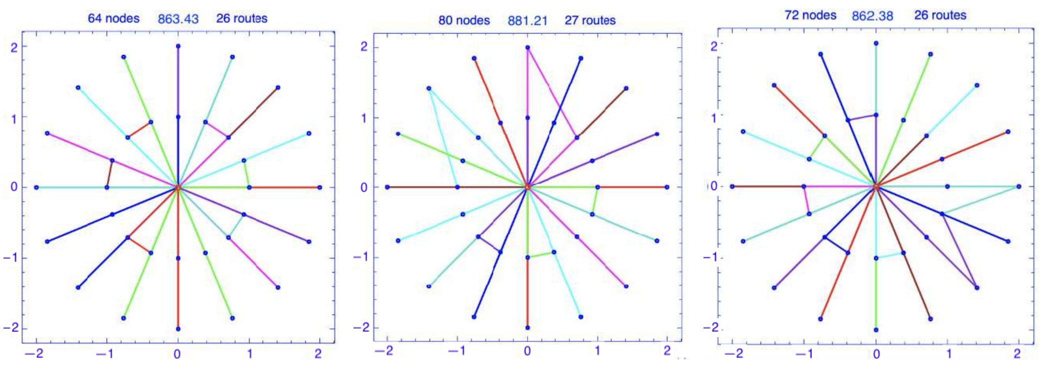
Impact of coarser rules for customers with higher demands: Using coarser rules for customers with high demands introduces a two-fold benefit: (i) it reduces the runtime due to a smaller number of resulting customers in the resulting CVRP and (ii) it often results in a lower total cost of delivery. To capture this effect, we employ an exponential pattern (such as 128/64/32/16/8/4/2) for the AASA rule. We illustrate this effect for the *P*04_3070 instance of the third dataset provided in [33], as depicted in Figure 1. In this instance, the vehicle capacity is 200 and the depot is located at the location $(0,0)$ in Figure 1a,b. Each point in the plane corresponds to a customer. The routes of the vehicles in the obtained solution are denoted by different colors. Figure 1a,b show the solution of the SDVRP instance considering the 20/10/5/1 and 128/64/32/16/8/2 splitting rules, respectively. We observe that the splitting rule of 128/64/32/16/8/4/2 reduces the solver runtime by an order of 2.5. The lower runtime is expected since using an exponential pattern in the splitting rule results in splitting the higher demands into a piece with a very high value and most likely fewer pieces in total. This effect can also be observed from the size of the generated CVRP instance. The 20/10/5/1 rule resulted in a new instance with 855 customers (nodes), while the 128/64/32/16/8/4/2 rule generated a network with 572 customers. Moreover, the coarser splitting rule of 128/64/32/16/8/4/2 resulted in a cost of 4424.64 that improves upon the cost of the rule 20/10/5/1 (proposed by the VRPHAS algorithm) by 1.34%.



(**a**) 20/10/5/1 rule, 231.91 s (**b**) 128/64/32/16/8/4/2 rule, 86.67 s

Figure 1. Impact of using splitting rules with different granularity for different demands.

Impact of coarser rules for customers further away from the depot: The second motivation behind our proposed AASA rule is that using coarser rules for the demands with a larger distance from the depot and splitting closer demands into smaller pieces results in lower solver runtime and possibly a better solution. The intuition behind this observation is that as the distance between a customer and the depot increases, each visit to that customer would significantly increase the cost. Therefore, it is desirable to satisfy the demand of further customers within the fewest possible number of visits and that is only possible by applying coarser splitting rules to those nodes. Moreover, by splitting closer-to-depot customer demands into smaller pieces, we provide the opportunity for the vehicles returning to the depot to utilize their remaining capacity to fulfill smaller demands and, therefore, minimize the fraction of their unused capacity. It is important to note that all of these intuitive statements are based on the fact that after splitting demands and generating a new network with a larger number of customers, we use the VRPH solver and basically solve the generated instance by a number of heuristic and metaheuristic methods. Figure 2a–c illustrate the above observation for the *SD*6 instance from [34]. The best-known solution for this instance is 830.86. In Figure 2a,b, the results of using the 80/40/20/10 and 40/20/10 rules are shown. Although these two rules alone provide acceptable results with the costs of 863.43 and 881.21, respectively, we observe that by using a combination of the two rules as shown in Figure 2c, we can achieve lower cost of 862.38. In this experiment, the 32 customers are clustered into two categories of size 16. For the 16 customers closer to the depot, we used the splitting rule 40/20/10, and the coarser splitting rule 80/40/20/10 is applied to the remaining 16 customers located farthest from the depot.



(**a**) 80/40/20/10 rule, 4.98 s (**b**) 40/20/10 rule, 5.90 s. (**c**) 80/40/20/10 and 40/20/10 combined rule, 5.41 s

Figure 2. Impact of using splitting rules with different granularity according to the customers' distance to the depot.

2.3.2. AASA Algorithm

The above observations highlight the significance of applying an adaptive splitting rule for different customers instead of using a fixed splitting rule for all customers, as in VRPHAS. Motivated by these observations, we incorporate the following factors into AASA:

- Vehicle capacity: As implied by [30], it makes sense to look for the splitting rule as a function of the vehicle capacity. This not only presents a structured representation with vehicle capacity as the kernel allowing for better generalization but also leads the policy towards leaving the lowest portion of the vehicles' capacities unused, and consequently, increases the vehicles' utilization.
- Customer location: We believe one potential improvement to the VRPHAS splitting rule is to personalize the rule for each customer considering its distance from the depot. We observe that if the customers that are located further from the depot are visited by one (or very few) vehicle(s), the total travel cost is lower. Otherwise, if a long distance needs to be traversed multiple times, the cost will be negatively impacted. This can be avoided by coarsening the splitting rule for further customers. In AASA, the impact of customers' locations is addressed by partitioning the customers into a number of clusters based on their distance from the depot. We then use a different splitting rule for each cluster. A coarser splitting rule is used for clusters of customers further from the depot while finer rules are applied to the clusters closer to the depot.
- Customer demand: Another important factor in the adaptation of the splitting rule is the demand of customers. Depending on how large the demand is, the splitting rule can be applied in full or partially to result in an appropriate number of additional demand points with appropriate demand values. In order to create the exponential pattern in the splitting rule, we propose to decompose the customers' demands based on the different powers of prime numbers. For instance, the demand of a customer with $d = 199$ served by vehicles with capacity $Q = 200$ can be decomposed as $d = 2^7 + 2^6 + 2^2 + 2^1 + 1$ if $p = 2$, and $d = 2 \times 3^4 + 3^3 + 3^2 + 1$ if $p = 3$.

Leveraging the above factors, the AASA solution framework is shown in Figure 3. First, AASA employs a clustering algorithm to partition the customers into multiple levels. In our experiments, we used a simple distance-based clustering as follows:

$$label(v) = \ell, \quad \text{if} \quad \frac{\ell - 1}{L} dist_{max} < dist(v, 0) \le \frac{\ell}{L} dist_{max}, \quad \ell \in \{1, \ldots, L\} \tag{5}$$

where L is the number of levels, and $dist_{max}$ is the maximum customer–depot distance among all customers, i.e, $dist_{max} = max_v\ dist(v,0)$. As a result of this clustering rule, first, the customers are separated with L uniformly spaced rings around the depot. Second, for the customers at each level, the same splitting rule is applied. After all the demands are split, a CVRP instance is generated. Finally, the VRPH solver is used to solve the resulting CVRP instance.

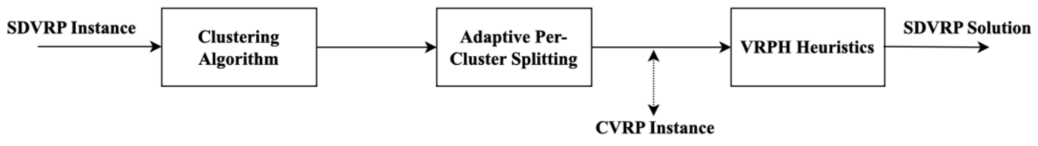


Figure 3. AASA framework.

The pseudo-code for AASA is given in Algorithm 1. The algorithm starts by taking as input the SDVRP instance as a graph $G = (V,E)$, the vehicle capacity Q, the vector of customer–depot distances denoted by $\vec{dist}$ where $\vec{dist}(v) = dist(v,0)$, and the vector of customer demands $\vec{D}$. Next, it computes d, the greatest common divisor (gcd) of the demands vector and the vehicle capacity, which will be used as a parameter in the adaptive splitting rule. The vehicles may carry goods only in quantities that are multiples of d units. This is to control the granularity of the splitting rule. Next, the demand vector is scaled down by d and averaged over the customers to find μ that is used in finding s_{max} determining the largest quantity of goods that is demanded by the customers in the equivalent CVRP model.

Algorithm 1 A priori Adaptive Splitting Algorithm ($AASA$)

```
Input: $G = (V,E), Q, \vec{dist}, \vec{D}, p$
Output: $G' = (V', E')$
 1: Initialize $L, d \leftarrow gcd(Q, \vec{D}), \mu \leftarrow avg(\vec{D}/d), s_{max} \leftarrow [log_p^{\mu}]$
 2: for $i$ : 1..$L$
 3:     $\vec{s_i} \leftarrow d * (exp_p\{0..s_{max} - i + 1\})$
 4: end for
 5: $\vec{label} = clustering(V)$                          $\triangleright\ \vec{label}(v) \in \{1, ..., L\}$
 6: repeat
 7:     $V = V \setminus \{v\}$
 8:     $i \leftarrow \vec{label}(v)$
 9:     $V' = V' \cup split(v, \vec{s_i})$
10: until $V = \emptyset$
11: return $G' = (V', E')$
```

The algorithm takes into account L different levels for the distance-based clustering as provided in (5). Then, for each level i, the splitting rule $\vec{s_i}$ is determined using the value s_{max}. Next, the clustering of the customers is made according to Equation (5). The *split* function substitutes each node (customer) of the graph G with new customers according to the rule $\vec{s_i}$ to gradually form the graph G'. Note that the splitting rule ensures that the demands further away from the depot are split with coarser rules. Once the graph G' is completed, the CVRP solver will solve the generated new problem. We experimented with different values of the prime number p and observed that $p = 2$ has the best performance overall in the current version of AASA.

For example, consider the instance $SD6$ from SET-1 as shown in Figure 2. We will have $d = gcd(\vec{D}, Q)$, i.e., $d = gcd(60, 90, 10) = 10$. The simple distance-based clustering in Equation (5) with $L = 2$ results in two levels, where $l = 1$ covers the customers that are furthest from the depot and $l = 2$ covers the customers that are closest to the depot. We will have $\mu = 7.5$ and $s_{max} = 3$. Therefore, for the first level, we have $\vec{s_1} = \{10, 20, 40, 80\}$ and for $\vec{s_2} = \{10, 20, 40\}$.

VRPH solver: Once the a priori splitting rule is applied to an SDVRP problem, any commercial solver can be used to solve the resulting CVRP instance. In this paper, we use the VRPH solver to solve CVRP instances. VRPH is a publicly available solver with a provable record of generating high-quality solutions. It takes as input a CVRP instance written to a file with a format similar to that of the Traveling Salesman Problem Library (TSPLIB) files and prints the solution to an output file. VRPH implements an open-source library of several local search heuristics for generating and improving feasible solutions to the CVRP instances. We refer the reader to [35] for a detailed description of the heuristics and the modular structure of the VRPH software. In this paper, we use VRPH as a standalone solver with the parameters set at their default values.

3. Performance Evaluation

In this section, we evaluate the performance of AASA through extensive numerical simulations. We benchmark the performance of AASA against three baseline strategies, using instances from datasets that are widely used in the literature for the comparison of various vehicle routing solutions. Table 1 presents the summary of the evaluation datasets used throughout this section. Further details about the instances of each dataset are provided in Section 3.1.

Table 1. Evaluation datasets.

Dataset	Number of Instances	Number of Customers (N)	Vehicle Capacity (Q)	Customers' Demands
SET-1	21	$[8, 288]$	100	$\{60, 90\}$
SET-2	14	$\{50, 75, 100\}$	160	randomly from $[aQ, bQ]$ [1]
SET-3	42	$[50, 199]$	$[140, 200]$	randomly from $[aQ, bQ]$ [1]
SET-4	11	$[21, 100]$	$[112, 8000]$	No pattern

[1] $(a, b) \in \{(0.01, 0.1), (0.1, 0.3), (0.1, 0.5), (0.1, 0.9), (0.3, 0.7), (0.7, 0.9)\}$, corresponding to six different cases.

3.1. Benchmarking Instances

We consider 4 benchmarking sets that include 88 instances overall as presented in Table 1. SET-1 [34] contains 21 instances, each with customers uniformly distributed on the perimeter of concentric circles centered at the depot. The instances are sorted by the number of customers and the number of concentric circles. SET-2 [36] and SET-3 [33] instances use similar customer demand profiles but vary in the number of customers and vehicle capacity. These demands are randomly sampled from the range $[aQ, bQ]$ with (a, b) chosen by six different scenarios as in Table 1. SET-2 contains 14 and SET-3 contains 42 instances, 6 of which are repeated. Therefore, we test only the 36 of them that are not redundant. The coordinates of the customers in SET-2 are randomly generated using the coordinates of eil51, eil76, and eil101 from TSPLIB, which are also used in SET-4 [37]. SET-4 includes 11 instances where no specific rule is preserved throughout for generating the customer demands.

3.2. Benchmarking Solutions

We benchmark the performance of the AASA algorithm against the following baseline methods:

- No-splitting (CVRP): The baseline method that treats an SDVRP instance as a CVRP instance and runs the VRPH [32] solver to generate solutions.
- VRPHAS: The method based on a priori splitting that is proposed by [30].
- ILS-MIP: The metaheuristic method proposed by [27].

We note that when computing the optimality gap corresponding to each method for each experimental instance, we take the best solution known in the literature as the reference solution. Furthermore, we note that in ILS-MIP, the travel cost between each pair of customers is assumed to be their rounded Euclidean distance except for SET-1 instances where the exact Euclidean distance is taken as the cost. Hence, we compare the results of ILS-MIP and AASA only for SET-1. For the rest of the datasets, we compare the performance of AASA against the No-splitting and the VRPHS methods.

3.3. Metrics and Setup

We consider the optimality gap and the solver runtime as metrics for performance comparison between the above strategies. The optimality gap which indicates the deviation from the best-known solution is defined as:

$$gap = \frac{obj - \text{best-known solution}}{\text{best-known solution}} * 100$$

where obj is the objective value of the considered method. We run both VRPHAS and AASA algorithms on the same PC with an Intel Xeon processor at 3.2 GHz and 16 GB of main memory. It is important to note that the gap results we obtained for the VRPHAS method do not match the values presented in [30] for some instances. However, since we need to compare both the runtime and the gap results for the VRPHAS and AASA methods on similar hardware, we reflect the results of our run for VRPHAS. Moreover, we report the computation time of the ILS-MIP method presented in [27], including the values of time and $\text{time}_{\text{best}}$ corresponding to the time required for the proposed iterative algorithm to terminate and the time at which the best solution of ILS-MIP is achieved, respectively. A time limit (TL) is considered for the algorithm termination that is assumed to be equal to 1349 s by [27]. It is important to note that the ILS experiments are conducted on a PC with an Intel Core i7-8700 3.2 GHz processor and 32 GB of memory which is more powerful than our PC. Therefore, we can expect that ILS-MIP performs even worse in terms of computation time compared to AASA if both were run on the same hardware.

3.4. Numerical Results

3.4.1. CVRP Instance Attributes Comparison: AASA vs. VRPHAS

As illustrated in Section 2.3.1, one of the promises of our proposed splitting algorithm is generating smaller size CVRP instances that can be solved consistently faster. To show this, we present the size of the CVRP instances and the number of routes (i.e., vehicles) in the generated CVRP solution obtained by the splitting strategies of VRPHAS and AASA for the evaluation sets SET-1 to SET-4 in Tables 2–5, respectively. It is observed that for all the benchmarking datasets, AASA reduces the size of the CVRP problem compared to the VRPHAS method. Moreover, the number of routes in the solution obtained by AASA is less than or equal to the VRPHAS solution except for a few instances. Although the reduced number of routes indicates the potential of AASA in producing better solutions compared to VRPHAS, it does not provide sufficient evidence for cost-efficiency since the actual routes taken by the vehicles determine the total cost. However, together with the solution gap results shown in the next section, we can conclude that AASA performs better than VRPHAS in terms of both solution quality and runtime and that is a result of the adaptive splitting approach we proposed.

Table 2. CVRP instance attributes: VRPHAS vs. AASA on SET-1.

Instance	**CVRP Size**		**Number of Routes**	
	VRPHAS	**AASA**	**VRPHAS**	**AASA**
SD1	32	18	8	7
SD2	64	42	14	13
SD3	64	36	13	13
SD4	96	54	19	20
SD5	128	84	26	26
SD6	128	72	29	28
SD7	160	110	34	34
SD8	192	134	41	41
SD9	192	126	39	39
SD10	256	168	55	56
SD11	320	222	68	68
SD12	320	220	68	68
SD13	384	272	80	83
SD14	480	330	103	100
SD15	576	403	123	121
SD16	576	324	128	129
SD17	640	444	138	140
SD18	640	440	141	139
SD19	768	535	167	165
SD20	960	666	210	208
SD21	1152	756	259	259

Table 3. CVRP instance attributes: VRPHAS vs. AASA on SET-2.

Instance	**CVRP Size**		**Number of Routes**	
	VRPHAS	**AASA**	**VRPHAS**	**AASA**
S51D1	139	109	3	3
S51D2	167	150	10	10
S51D3	209	162	17	17
S51D4	246	247	32	32
S51D5	253	231	28	28
S51D6	307	246	50	50
S76D1	218	176	4	4
S76D2	263	245	16	16
S76D3	327	283	24	24
S76D4	352	352	41	40
S101D1	276	218	5	5
S101D2	345	315	21	21
S101D3	428	372	34	33
S101D5	500	472	58	56

Table 4. CVRP instance attributes: VRPHAS vs. AASA on SET-3.

Instance	**CVRP Size**		**Number of Routes**	
	VRPHAS	**AASA**	**VRPHAS**	**AASA**
p01_110	121	92	3	3
p01_1030	176	148	11	11
p01_1050	207	193	17	17
p01_1090	268	204	33	33
p01_3070	257	198	33	33
p01_7090	326	240	50	50
p02_110	182	168	5	5
p02_1030	204	231	17	17

Table 4. *Cont.*

Instance	CVRP Size		Number of Routes	
	VRPHAS	**AASA**	**VRPHAS**	**AASA**
p02_1050	297	283	27	27
p02_1090	382	377	48	47
p02_3070	370	293	48	48
p02_7090	462	377	75	75
p03_110	346	244	6	6
p03_1030	411	333	23	23
p03_1050	471	376	37	37
p03_1090	608	476	67	67
p03_3070	571	369	66	66
p03_7090	711	591	100	99
p04_110	516	315	10	10
p04_1030	613	467	34	34
p04_1050	705	547	55	55
p04_1090	915	676	98	98
p04_3070	855	649	98	98
p04_7090	1063	879	149	149
p05_110	657	475	12	12
p05_1030	804	616	43	43
p05_1050	920	621	67	69
p05_1090	1169	824	123	124
p05_3070	1119	806	124	126
p05_7090	1401	1082	199	197
p11_110	422	233	8	8
p11_1030	489	388	27	27
p11_1050	560	475	42	43
p11_1090	722	543	78	78
p11_3070	680	500	77	76
p11_7090	849	663	119	119

Table 5. CVRP instance attributes: VRPHAS vs. AASA on SET-4.

Instance	CVRP Size		Number of Routes	
	VRPHAS	**AASA**	**VRPHAS**	**AASA**
eil22	69	47	4	4
eil23	74	44	3	3
eil30	110	57	4	4
eil33	108	72	4	4
eil51	157	139	6	6
eilA76	228	215	10	10
eilB76	254	215	15	15
eilC76	231	201	8	8
eilD76	216	215	7	7
eilA101	347	248	8	8
eilB101	265	248	14	14

3.4.2. Optimality Gap and Runtime Comparison: AASA against Baselines

Table 6 presents the performance of AASA against VRPHAS, ILS-MIP, and the no-splitting strategies for the instances of SET-1. First, we observe that the no-splitting method results in very low-quality solutions as the resulting optimality gap obtained by this method is very high compared to the VRPHAS, AASA, and ILS-MIP heuristics. Moreover, we observe that while the ILS-MIP strategy outperforms the AASA and VRPHAS solution approaches in terms of optimality gap for almost all instances of SET-1, it requires very large computation times. The average runtime for ILS-MIP is 992.38 s as opposed to the average runtime of 86.14 s and 30.56 s corresponding to the VRPHAS and AASA solution approaches, respectively. This huge time complexity hinders the practical application of the ILS-MIP approach in real-world problems with time sensitivity or scalability requirements. Figures 4 and 5 illustrate the per-instance optimality gap and runtime of our proposed splitting rule against the fixed splitting rule introduced in VRPHAS. Our proposed method outperforms the VRPHAS in terms of time complexity for all instances, and it also improves upon the average optimality gap of VRPHAS. Moreover, while the VRPHAS method results in the 5.50% optimality gap in the worst-case scenario (*SD*16), AASA achieves the maximum optimality gap of 3.81% for the same instance. We also observe that AASA finds the best-known solution for four instances as opposed to the VRPHAS method obtaining the best-known solution for three instances of SET-1.

Table 6. Performance of VRPHAS, AASA, and ILS-MIP on SET-1.

Instance	Best-Known	VRPHAS		AASA			ILS-MIP		No Splitting
	Solution	Gap (%)	Time (s)	Cost	Gap (%)	Time (s)	Gap	Time –Time$_{best}$ (s)	Gap (%)
SD1	228.28	0.00	0.51	228.28	0.00	1.60	0.00	25.10–0.00	5.13
SD2	708.28	0.00	1.23	708.28	0.00	2.85	0.00	47.51–0.00	12.94
SD3	430.40	0.04	1.25	430.58	0.04	2.57	0.05	37.68–0.00	11.51
SD4	630.62	0.07	2.35	631.04	0.07	3.57	0.07	75.45–0.00	14.17
SD5	1389.94	1.22	3.91	1390.57	0.05	5.66	0.05	216.69–0.80	15.10
SD6	830.86	0.05	3.85	831.24	0.05	5.09	0.04	202.60–92.42	15.47
SD7	3640.00	0.55	5.47	3640.00	0.00	7.79	0.00	TL–0.00	20.87
SD8	5068.28	0.00	7.64	5068.28	0.00	10.43	0.00	TL–0.00	23.11
SD9	2042.88	0.72	7.50	2051.06	0.40	8.97	0.07	TL–39187	17.47
SD10	2683.73	0.95	13.73	2704.88	0.79	13.68	0.38	TL–960.42	19.23
SD11	13,280.00	0.15	20.59	13,480.00	1.50	18.43	0.00	TL–0.00	26.50
SD12	7213.62	0.73	20.89	7238.88	0.34	20.99	0.09	TL–1223.31	21.99
SD13	10,105.86	0.05	25.90	10,110.58	0.05	22.05	0.05	TL–1091.84	23.49
SD14	10,717.53	0.70	42.57	10,804.84	0.81	32.03	0.22	TL–967.30	23.16
SD15	15,094.48	0.93	59.43	15,232.60	0.91	47.32	0.37	TL–318.44	24.01
SD16	3379.33	5.50	54.88	3508.16	3.81	30.16	0.11	TL–0.01	27.83
SD17	26,493.56	0.25	66.98	26,962.64	1.56	50.14	0.1	TL–1328.05	26.82
SD18	14,202.53	0.69	72.04	14,278.70	0.54	54.75	0.46	TL–1347.05	23.92
SD19	19,995.69	1.00	101.05	20,197.04	1.00	70.14	0.64	TL–1328.17	24.82
SD20	39,635.51	0.17	162.90	40,236.66	1.51	104.51	0.25	TL–1328.44	27.15
SD21	11,271.06	3.41	240.36	11,576.96	2.71	129.31	0.2	TL–0.07	27.76
Average		0.82	86.14		0.76	30.56	0.15	992.38–494.17	20.60

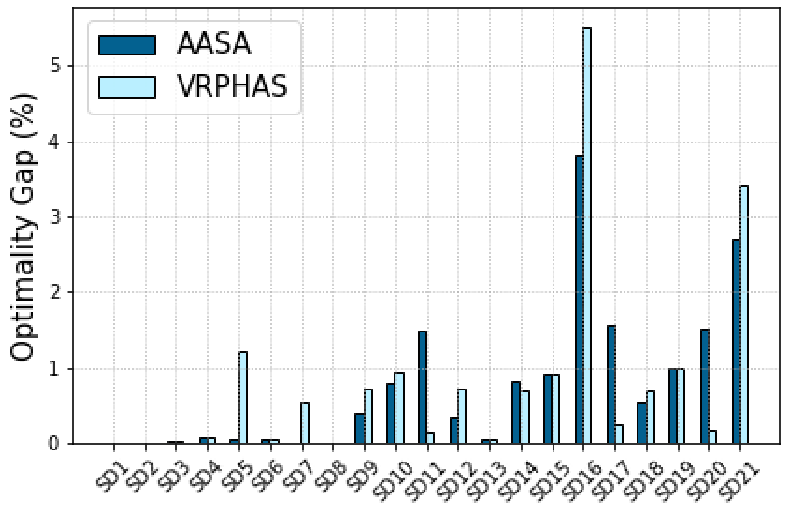


Figure 4. Per-instance gap of SET-1.

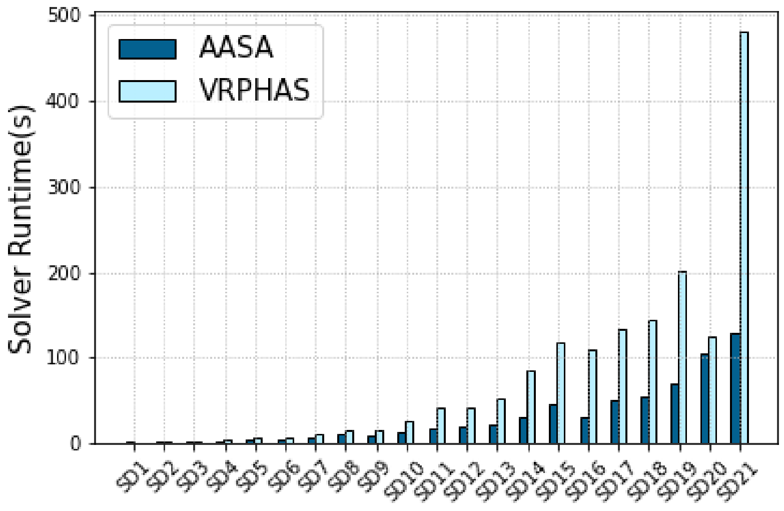


Figure 5. Per-Instance solver runtime of SET-1.

Table 7 shows the results of VRPHAS, AASA, and the no-splitting methodologies for the instances of SET-2. According to Table 7, the no-splitting approach results in a higher gap compared to both the VRPHAS and AASA rules. Furthermore, our proposed AASA strategy outperforms the VRPHAS method in terms of the average optimality gap, as well as runtime. Moreover, the per-instance results depicted in Figures 6 and 7 indicate that the AASA strategy improves upon the optimality gap and solver runtime of the VRPHAS method for almost all instances of SET-2. It can also be observed that while the VRPHAS method exhibits an optimality gap of 2.51% in the worst-case scenario ($S76D4$), AASA achieves a maximum optimality gap of 1.97% ($S51D6$).

Table 7. Performance of VRPHAS, AASA, and the no-splitting methodologies on SET-2.

Instance	Best-Known	VRPHAS		AASA			No Splitting
	Solution	Gap (%)	Time (s)	Cost	Gap (%)	Time (s)	Gap (%)
S51D1	459.50	0.00	10.98	459.50	0.00	7.92	0.00
S51D2	709.29	0.77	15.04	713.80	0.63	11.22	1.71
S51D3	948.06	1.88	20.28	959.92	1.25	13.04	3.19
S51D4	1562.01	1.94	28.24	1583.88	1.40	22.94	7.32
S51D5	1333.67	1.03	30.68	1343.109	0.71	20.81	8.89
S51D6	2169.10	1.86	43.81	2211.90	1.97	23.37	10.75
S76D1	598.94	0.00	23.18	598.94	0.00	14.55	0.78
S76D2	1087.40	1.82	32.48	1104.88	1.61	22.43	2.79
S76D3	1427.86	1.93	46.99	1436.99	0.64	28.05	2.52
S76D4	2079.76	2.51	55.89	2120.16	1.94	40.09	5.41
S101D1	726.59	1.23	34.03	729.72	0.43	42.31	1.14
S101D2	1378.43	2.00	52.44	1399.67	1.54	34.45	1.67
S101D3	1874.81	1.85	70.64	1905.43	1.63	41.05	4.13
S101D5	2791.22	1.61	95.71	2844.51	1.91	62.54	11.59
Average		1.39	39.94		1.08	26.12	4.42

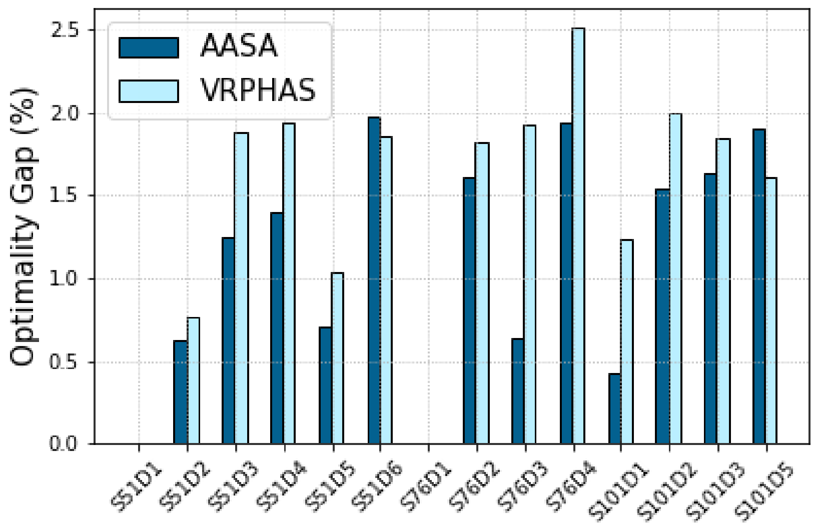


Figure 6. Per-instance gap of SET-2.

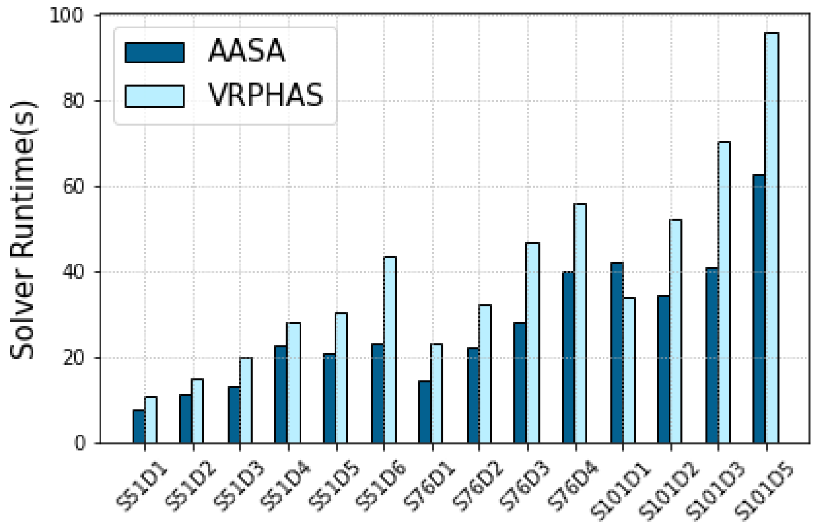


Figure 7. Per-instance solver runtime of SET-2.

Similarly, the comparison of the results for SET-3 and SET-4 provided in Tables 8 and 9 show that our proposed splitting algorithm again improves both average optimality gap and runtime of the VRPHAS splitting mechanism and, therefore, it can effectively generate higher-quality solutions, faster than VRPHAS. The improvement in the runtime is mainly owing to the rule granularity consideration introduced in AASA as discussed in Section 2.2, which results in smaller instances to be solved by the CVRP solver compared to the VRPHAS method. Figures 8 and 9 illustrate the per-instance optimality gap and runtime of SET-3. According to Table 8 and Figure 8, although the VRPHAS method results in the maximum optimality gap of 3.27% (p11_1090), the worst-case optimality gap achieved by AASA is 2.54% (p02_7090). The optimality gap and run-time of SET-4 instances are presented in Figures 10 and 11. Table 9 and Figure 10 illustrate that the AASA method improves the worst-case optimality gap of VRPHAS in SET-4, since AASA achieves the maximum optimality gap of 2.02% ($eilA$101) as opposed to the maximum gap of 2.35% obtained by VRPHAS for the same instance.

Table 8. Performance of VRPHAS, AASA, and the no-splitting methodologies on SET-3.

Instance	Best-Known	VRPHAS		AASA			No Splitting
	Solution	Gap (%)	Time (s)	Cost	Gap (%)	Time (s)	Gap (%)
p01_110	459.50	0.00	9.84	459.50	0.00	6.63	0.00
p01_1030	757.17	1.74	17.2	770.93	1.81	11.08	3.29
p01_1050	1005.75	2.56	22.52	1021.18	1.53	15.76	4.08
p01_1090	1488.58	1.48	32.70	1507.14	1.24	19.61	12.75
p01_3070	1481.71	2.23	32.84	1503.66	1.48	17.02	13.81
p01_7090	2156.14	2.24	49.82	2203.93	2.22	21.41	11.41
p02_110	617.85	0.66	21.82	618.13	0.04	12.08	0.65
p02_1030	1109.62	1.65	28.86	1125.65	1.44	21.83	2.50
p02_1050	1502.05	1.06	22.52	1508.98	0.46	26.21	2.72
p02_1090	2298.58	2.00	61.01	2334.33	1.55	41.19	9.21
p02_3070	2219.97	2.23	58.80	2269.50	2.23	28.02	12.87
p02_7090	3223.4	1.57	89.76	3305.53	2.54	40.52	12.64
p03_110	752.62	1.78	55.12	762.54	1.31	22.08	1.51
p03_1030	1458.46	1.87	37.26	1476.86	1.26	35.67	2.09
p03_1050	1996.76	2.54	58.40	2032.25	1.77	41.76	3.93
p03_1090	3085.69	2.29	143.32	3143.69	1.88	56.56	12.17
p03_3070	2989.3	2.19	136.10	3034.28	1.50	40.02	16.50
p03_7090	4387.32	1.66	186.14	4464.65	1.76	84.48	13.72
p04_110	919.17	1.35	116.08	925.19	0.65	33.83	0.84
p04_1030	2016.97	2.71	152.44	2056.71	1.97	55.37	2.97
p04_1050	2849.66	2.40	196.82	2889.62	1.40	72.14	5.09
p04_1090	4545.46	2.44	310.84	4624.18	1.73	112.41	14.72
p04_3070	4334.71	2.87	272.44	4407.02	1.67	99.65	19.64
p04_7090	6395.41	1.94	469.62	6555.47	2.5	82.07	15.09
p05_110	1074.18	2.74	186.66	1086.27	1.12	66.5	1.42
p05_1030	2478.4	1.83	241.44	2515.45	1.49	82.72	2.46
p05_1050	3471.41	1.54	311.94	3510.99	1.14	87.56	4.01
p05_1090	5521.57	2.02	561.54	5619.52	1.77	145.02	12.67
p05_3070	5409.76	2.41	383.28	5412.85	0.06	142.81	15.57
p05_7090	8192.03	1.96	884.68	8332.36	1.71	255.05	17.28
p11_110	1031.11	2.11	74.12	1044.44	1.29	20.20	1.55
p11_1030	2881.8	2.89	94.06	2923.86	1.46	40.64	1.95
p11_1050	4219.01	2.29	108.54	4268.79	1.18	55.27	2.71
p11_1090	6854.09	3.27	188.30	6994.56	2.05	64.06	13.22
p11_3070	6671.04	1.44	162.36	6759.99	1.33	59.01	15.79
p11_7090	10,204.81	1.99	231.60	10,406.25	1.97	101.02	19.93
Average		1.96	166.94		1.45	33.38	8.08

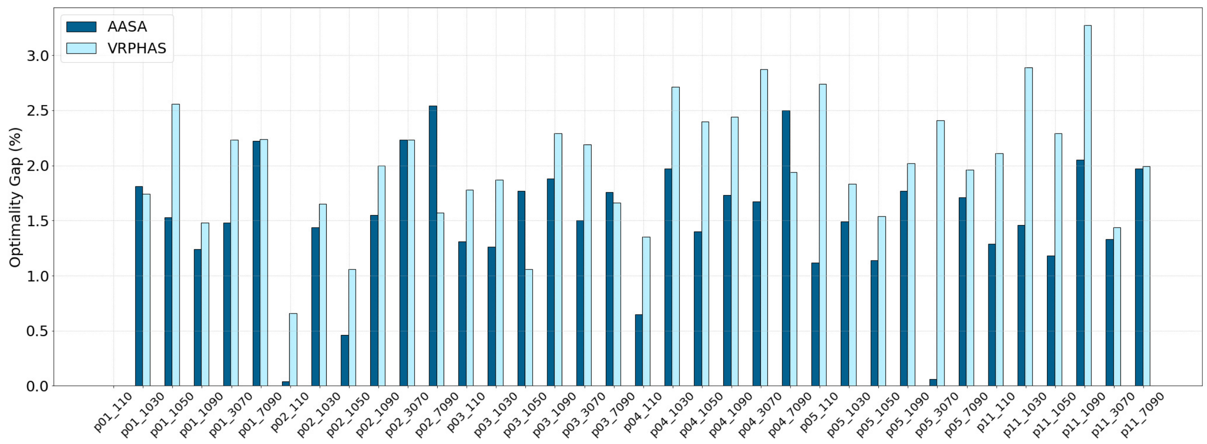


Figure 8. Per-instance gap of SET-3.

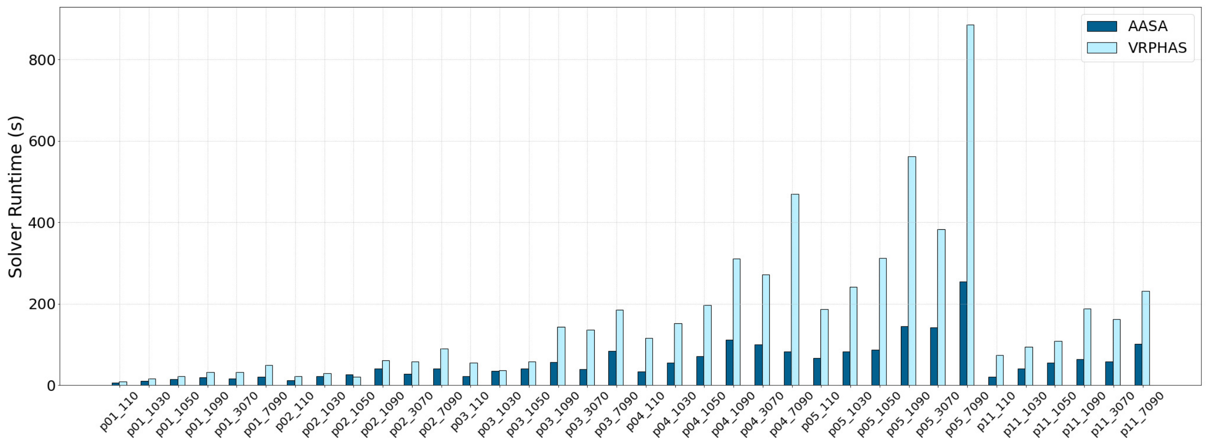


Figure 9. Per-instance solver runtime of SET-3.

Table 9. Performance of VRPHAS, AASA, and the no-splitting methodologies on SET-4.

Instance	Best-Known	VRPHAS		AASA			No Splitting
	Solution	Gap (%)	Time (s)	Cost	Gap (%)	Time (s)	Gap (%)
eil22	375.28	0.00	3.28	375.28	0.00	3.11	0.00
eil23	568.56	0.00	5.92	568.56	0.00	2.92	0.00
eil30	497.53	1.50	6.89	505.01	1.50	3.64	1.48
eil33	826.41	1.36	6.89	837.67	1.36	5.08	1.36
eil51	524.61	0.00	12.58	524.61	0.00	10.95	0.00
eilA76	823.89	0.43	24.60	830.97	0.86	19.77	1.61
eilB76	1009.04	1.96	31.86	1025.24	1.60	18.65	1.86
eilC76	738.67	0.52	26.78	742.49	0.55	17.48	0.64
eilD76	684.53	0.81	23.82	689.47	0.72	19.41	0.82
eilA101	812.51	2.35	55.39	828.98	2.02	23.37	1.92
eilB101	1076.26	1.57	30.90	1087.70	1.06	23.76	1.53
Average		0.96	20.80		0.88	13.63	1.02

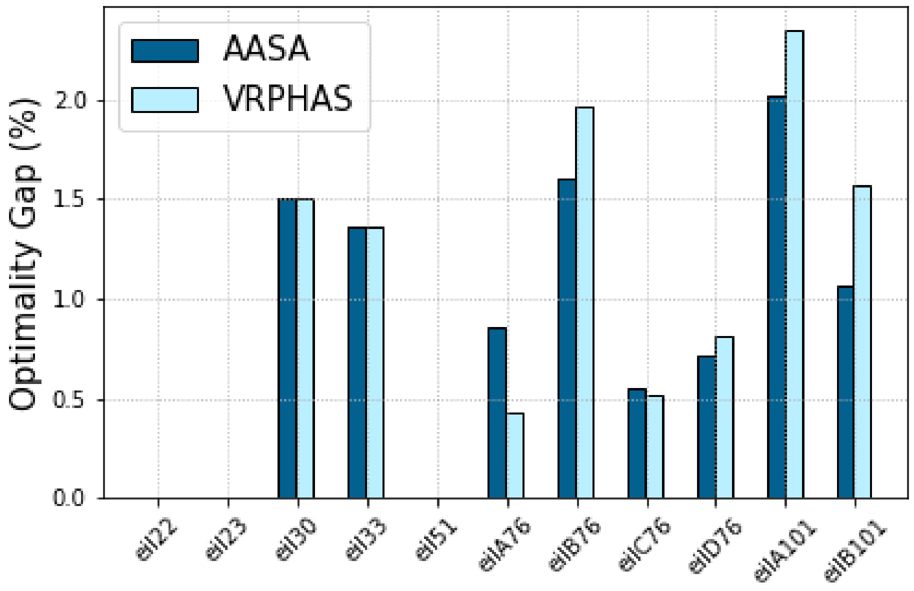


Figure 10. Per-instance AASA gap of SET-4.

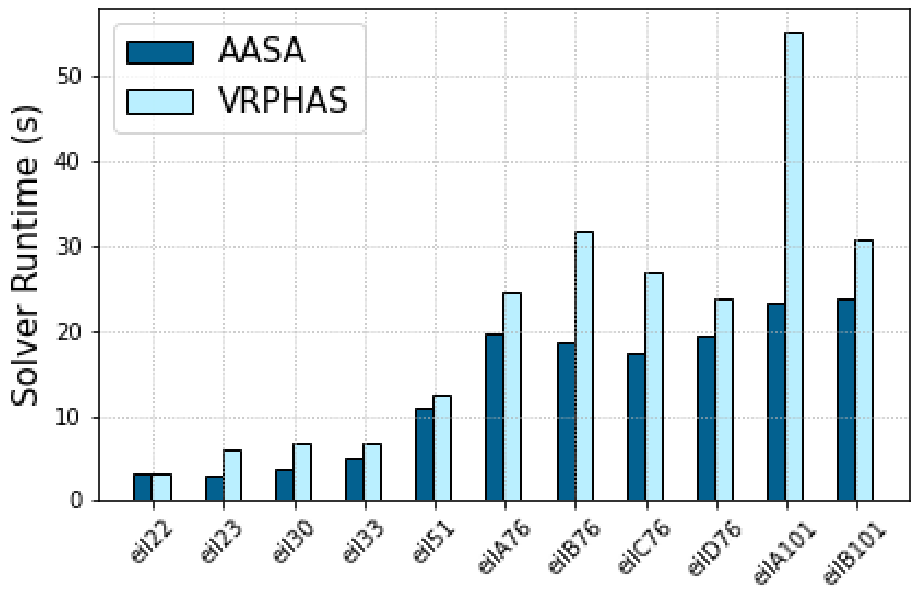


Figure 11. Per-instance solver runtime of SET-4.

4. Conclusions

In this paper, we propose an efficient and effective heuristic algorithm for the SDVRP based on a two-step procedure of splitting the demand of customers into smaller demands and solving the generated network assuming it to be a CVRP instance. Our approach focuses on enhancing the state-of-the-art splitting rule by considering customer locations and demand values. Our proposed adaptive splitting algorithm incorporates clustering customers based on their location and choosing a specific splitting rule for each cluster. It is shown to outperform the fixed splitting rule of the approach adopted by the state of the art, for all four benchmark datasets on average, with respect to both performance and computational effort.

Author Contributions: Conceptualization, N.T., A.G., J.S.B. and B.L.G.; Methodology, N.T., A.G. and J.S.B.; Software, N.T. and A.G.; Validation, A.G. and B.L.G.; Formal analysis, N.T., A.G. and B.L.G.; Investigation, N.T. and A.G.; Resources, N.T. and A.G.; Writing – original draft, N.T., A.G. and B.L.G.; Writing – review and editing, J.S.B. and B.L.G.; Visualization, J.S.B. and B.L.G.; Supervision, J.S.B. and B.L.G.; Project administration, J.S.B.; Funding acquisition, N.T., A.G. and J.S.B. All authors have read and agreed to the published version of the manuscript.

Funding: The research of N. Torkzaban, A. Gholami, and J. Baras was partially supported by ONR grant N00014-17-1-2622, and by a grant from Lockheed Martin Chair in Systems Engineering.

Data Availability Statement: Data are contained within the article.

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

References

1. Dror, M.; Trudeau, P. Savings by split delivery routing. *Transp. Sci.* **1989**, *23*, 141–145. [CrossRef]
2. Archetti, C.; Savelsbergh, M.; Speranza, M. Worst-case analysis for split delivery vehicle routing problems. *Transp. Sci.* **2006**, *40*, 226–234. [CrossRef]
3. Mullaseril, P.; Dror, M.; Leung, J. Split-delivery routing heuristics in livestock feed distribution. *J. Oper. Res. Soc.* **1997**, *48*, 107–116. [CrossRef]
4. Sierksma, G.; Tijssen, G. Routing helicopters for crew exchanges on off-shore locations. *Ann. Oper. Res.* **1998**, *76*, 261–286. [CrossRef]
5. Song, S.; Lee, K.; Kim, G. A practical approach to solving a newspaper logistics problem using a digital map. *Comput. Ind. Eng.* **2002**, *43*, 315–330. [CrossRef]
6. Luo, K.; Zhao, W.; Zhang, R. A multi-day waste collection and transportation problem with selective collection and split delivery. *Appl. Math. Model.* **2024**, *126*, 753–771. [CrossRef]

7. Archetti, C.; Feillet, D.; Gendreau, M.; Speranza, M.G. Complexity of the VRP and SDVRP. *Transp. Res. Part C Emerg. Technol.* **2011**, *19*, 741–750. [CrossRef]
8. Archetti, C.; Bianchessi, N.; Speranza, M.G. A column generation approach for the split delivery vehicle routing problem. *Networks* **2011**, *58*, 241–254. [CrossRef]
9. Hernández-Pérez, H.; Salazar-González, J.-J. A Branch-and-cut algorithm for the split-demand one-commodity pickup-anddelivery travelling salesman problem. *Eur. J. Oper. Res.* **2021**, *297*, 467–483. [CrossRef]
10. Wolfinger, D.; Salazar-González, J.-J. The pickup and delivery problem with split loads and transshipments: A branch-and-cut solution approach. *Eur. J. Oper. Res.* **2020**, *289*, 470–484. [CrossRef]
11. Casazza, M.; Ceselli, A.; Calvo, R.W. A route decomposition approach for the single commodity Split Pickup and Split Delivery Vehicle Routing Problem. *Eur. J. Oper. Res.* **2019**, *289*, 897–911. [CrossRef]
12. Li, J.L.; Qin, H.; Shen, H.X.; Tong, X.; Xu, Z. Exact algorithms for the multiple depot vehicle scheduling problem with heterogeneous vehicles, split loads and toll-by-weight scheme. *Comput. Ind. Eng.* **2022**, *168*, 108137. [CrossRef]
13. Li, J.; Qin, H.; Baldacci, R.; Zhu, W. Branch-and-price-and-cut for the synchronized vehicle routing problem with split delivery, proportional service time and multiple time windows. *Transp. Res. Part E Logist. Transp. Rev.* **2020**, *140*, 101955. [CrossRef]
14. Gschwind, T.; Bianchessi, N.; Irnich, S. Stabilized branch-price-and-cut for the commodity-constrained split delivery vehicle routing problem. *Eur. J. Oper. Res.* **2019**, *278*, 91–104. [CrossRef]
15. Min, J.N.; Jin, C.; Lu, L.J. Maximum-minimum distance clustering method for split-delivery vehicle-routing problem: Case studies and performance comparisons. *Adv. Prod. Eng. Manag.* **2019**, *14*, 125–135. [CrossRef]
16. Wang, J.; Jagannathan, A.K.; Zuo, X.; Murray, C.C. Two-layer simulated annealing and tabu search heuristics for a vehicle routing problem with cross docks and split deliveries. *Comput. Ind. Eng.* **2017**, *112*, 84–98. [CrossRef]
17. Xia, Y.; Fu, Z. An adaptive tabu search algorithm for the open vehicle routing problem with split deliveries by order. *Wirel. Pers. Commun.* **2018**, *103*, 595–609. [CrossRef]
18. Xia, Y.; Fu, Z.; Tsai, S.B.; Wang, J. A new TS algorithm for solving low-carbon logistics vehicle routing problem with split deliveries by backpack—From a green operation perspective. *Int. J. Environ. Res. Public Health* **2018**, *15*, 949. [CrossRef] [PubMed]
19. Ma, X.; Liu, C. Improved Ant Colony Algorithm for the Split Delivery Vehicle Routing Problem. *Appl. Sci.* **2024**, *14*, 5090. [CrossRef]
20. Yang, W.; Wang, D.; Pang, W.; Tan, A.-H.; Zhou, Y. Goods consumed during transit in split delivery vehicle routing problems: Modeling and solution. *IEEE Access* **2020**, *8*, 110336–110350. [CrossRef]
21. Jiang, Y.; Bian, B.; Liu, Y. Integrated multi-item packaging and vehicle routing with split delivery problem for fresh agri-product emergency supply at large-scale epidemic disease context. *J. Traffic Transp. Eng. (Engl. Ed.)* **2020**, *8*, 196–208. [CrossRef]
22. Fan, H.M.; Zhang, X.; Ren, X.X.; Liu, P. Optimization of multi-depot open split delivery vehicle routing problem with simultaneous delivery and pick-up. *Syst. Eng.-Theory Pract.* **2021**, *41*, 1521–1534.
23. Shi, J.; Zhang, J.; Wang, K.; Fang, X. Particle swarm optimization for split delivery vehicle routing problem. *Asia-Pac. J. Oper. Res.* **2018**, *35*, 1840006. [CrossRef]
24. Qing, D.S.; Deng, Q.L.; Li, J.J.; Liu, S.; Liu, X.; Zeng, S.P. Split vehicle route planning with full load demand based on particle swarm optimization. *J. Control Decis.* **2021**, *36*, 1397–1406.
25. Jin, M.; Liu, K.; Eksioglu, B. A column generation approach for the split delivery vehicle routing problem. *Oper. Res. Lett.* **2008**, *36*, 265–270. [CrossRef]
26. Bianchessi, N.; Drexl, M.; Irnich, S. The split delivery vehicle routing problem with time windows and customer inconvenience constraints. *Transp. Sci.* **2019**, *53*, 1067–1084. [CrossRef]
27. Alvarez, A.; Munari, P. A Matheuristic Approach for Split Delivery Vehicle Routing Problems. 2022. Available online: http://www.optimization-online.org/DB_FILE/2022/02/8790.pdf (accessed on 31 August 2023).
28. Vidal, T. Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. *Comput. Oper. Res.* **2022**, *140*, 105643. [CrossRef]
29. Munari, P.; Savelsbergh, M. Compact formulations for split delivery routing problems. *Transp. Sci.* **2022**, *56*, 1022–1043. [CrossRef]
30. Chen, P.; Golden, B.; Wang, X.; Wasil, E. A novel approach to solve the split delivery vehicle routing problem. *Int. Trans. Oper. Res.* **2017**, *24*, 27–41. [CrossRef]
31. Salani, M.; Vacca, I. Branch and price for the vehicle routing problem with discrete split deliveries and time windows. *Eur. J. Oper. Res.* **2011**, *213*, 470–477. [CrossRef]
32. Groer, C. VRPH. 2011. Available online: https://github.com/coin-or/VRPH (accessed on 1 August 2024).
33. Archetti, C.; Speranza, M.G.; Savelsbergh, M.W. An optimization-based heuristic for the split delivery vehicle routing problem. *Transp. Sci.* **2008**, *42*, 22–31. [CrossRef]
34. Chen, S.; Golden, B.; Wasil, E. The split delivery vehicle routing problem: Applications, algorithms, test problems, and computational results. *Netw. Int. J.* **2007**, *49*, 318–329. [CrossRef]
35. Groer, C.S.; Golden, B.; Wasil, E. A Library of Local Search Heuristics for the Vehicle Routing Problem. *Math. Program. Comput.* **2010**, *2*, 79–101. [CrossRef]

36. Belenguer, J.M.; Martinez, M.C.; Mota, E. A lower bound for the split delivery vehicle routing problem. *Oper. Res.* **2000**, *48*, 801–810. [CrossRef]
37. Reinhelt, G. TSPLIB: A Library of Sample Instances for the TSP (and Related Problems) from Various Sources and of Various Types. 2014. Available online: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95 (accessed on 31 August 2023).