CIE50 Proceedings, October 30-November 2, 2023
American University of Sharjah, UAE

Smart and Parallel General Variable Neighborhood Search for the Pollution-Routing Problem

Mário Leite[a], Telmo Pinto[b,a], Cláudio Alves[b]

[a]*ALGORITMI Research Centre / LASI , University of Minho, Campus de Gualtar, 4710-057, Braga, Portugal*
[b]*University of Coimbra, CEMMPRE, ARISE, Paço das Escolas, 3004-531, Coimbra, Portugal*

Abstract

This work addresses a new approach to the Pollution-Routing Problem (PRP), a variant of the Vehicle Routing Problem (VRP) under environmental concerns, which includes costs associated with fuel, drivers, and greenhouse gas emissions. The many factors impacting the environment and, simultaneously, the real cost of the routes are usually ignored in the approaches defined to solve routing problems since the total distance traveled remains the standard objective. However, in pollution-routing problems, these elements play an essential role and each one is significantly influenced by the vehicle load and/or speed over the pathways which are followed. To contribute with methods that can provide solutions within an acceptable computational time, we explore local search and meta-heuristic based approaches, with emphasis on a Smart and Parallel General Variable Neighborhood Search algorithm for the PRP. Innovative neighborhood structures allowing continuous speed values in the arcs were a implemented. Additionally, we incorporate parallel programming strategies. To evaluate the effectiveness of these strategies, we report on the computational experiments conducted on benchmark instances, and we compare the results obtained with other studies from the literature.

Keywords: pollution-routing, variable neighborhood search, parallelism

1. INTRODUCTION

In recent decades, global warming and climate change have led to a growing awareness of environmental issues and the consequent need to modify existing procedures while becoming increasingly sustainable. In industry, and more precisely in transportation through heavy-duty vehicles, greenhouse gas emissions are one of the main impact factors with high amounts emitted [1, 2, 3].

However, in most cases, changes in supply chain processes to reduce emissions imply that strategies with environmental concerns are chosen in detriment of financial and profit aspects, which presents itself as an obstacle to their implementation in the companies. Therefore, the different world organisations define specific guidelines and regulations for companies, where they must commit to meeting the objectives to avoid sanctions. For instance, in 2011, the European Commission outlined cutting carbon emissions by 20% in the transport sector by 2030 and up to 60% by 2050 [4]. Some alternatives can be sought to reduce emissions in transport, such as having fleets with cleaner or more efficient fuels, trying to avoid congested and sloped roads, having drivers trained with skills to control the vehicle in all its extensions (as speed, acceleration rate, brake usage, shifting technique), seeking to ideally combine vehicle capacity with the loads to be transported, among others [5, 6].

In this context, the problem addressed in this work tried to reduce environmental impact in transportation and was initially introduced in 2011 in Bektaş and Laporte [7]. Until there, the common approaches to routing problems tend to consider the single objective of minimizing costs or the distance traveled (usually a linear function). However, in Bektaş and Laporte [7], the authors proposed Pollution-Routing Problem (PRP) as an extension of the Vehicle Routing Problem (VRP), which encloses greenhouse emissions, fuel, and driver wages.

The main problem features presented are as follows: only one depot is available (where routes should begin and finish); time windows in nodes (customers); a homogeneous fleet; customers with different demands and service times; and limited speeds on the arcs. Its objective is to minimize CO2 emissions, fuel consumption, and driver costs. In PRP, each element of the objective function significantly relies on the vehicle load and speed in each arc it travels, taking into account several factors studied in Barth et al. [8] and Barth and Boriboonsomsin [9] that influence the energy needed to move the heavy-duty diesel truck on a path. Among these factors, which remain constant in all arcs for practical reasons, other characteristics are considered, such as the road (road angle, air density), the characteristics of the vehicle itself (engine module, vehicle accessories, the mass of the empty vehicle, front surface area, coefficients of aerodynamic drag and rolling resistance), and the skills of the driver (vehicle drive train efficiency parameter). The amount of fuel consumption and emitted gases is proportionally obtained based on the approximation of the energy consumed in each arc traveled, with a specific load and an average speed. All these components conduct the objective function to a non-linear behavior. There are several counter-intuitive situations. Fewer load trips lead to fewer gas emissions, but less load in the vehicle conduct to more distance to travel (due to the need for more routes to transport all the demand, therefore more trips to perform) and, consequently, more emissions. A considerably lower speed, which at the outset could be seen as a good decision to reduce emissions, may also imply more distance to be covered (in this case, due to the high time to cover the routes and to arrive before the end of the customer time window, and then routes with few customers would be required). On the other hand, high speed leads to more fuel consumption. However, it also means less cost for drivers if time-dependent wages are considered.

According to the 2022 report from the European Environment Agency [10], there has been a reduction in emissions in the transport sector due to recent restrictions on the manufacture of new vehicles so that they are more efficient in fuel combustion (complying with Europe CO2 emission targets). This overall reduction is also due to the increase in registrations of zero- and low-emission vehicles (battery electric vehicles). The principles of the problem described in this work remain valid even when having an electric fleet. There is no concern with fuel consumption, but minimizing the energy needed to move the vehicle is still relevant, thus increasing the vehicle's autonomy (in this case, thus reducing energy consumption).

The Pollution-Routing Problem is an NP-hard problem. Exact methods are used to tackle the PRP [11, 12]. However, achieving solutions in most instances is challenging, even with few customers. For large-scale instances, exact methods are computationally intractable. Not surprisingly, some approaches rely on heuristic-based methods. In Demir et al. [13], the authors proposed a heuristic divided into two phases solved sequentially. First, through an Adaptive Large Neighborhood Search (ALNS) heuristic, with successive removal and insertion operators in the solution, a solution for a VRP with Time Windows is obtained by determining the order of visit of the nodes in each route and assuming a fixed initial speed in the arcs. Then, using the solution of the first stage and through a Speed Optimization Algorithm (SOA), the optimal speed of each arc traversed is determined for each route to minimize the objective of the PRP. This post-processed speed optimization, to occur only after the decision of the routes to be taken, is pointed out in Kramer et al. [14] as a problem that leads to lower-quality solutions. Consequently, to overcome this weakness, Kramer et al. [14] presents a hybrid algorithm that integrates an Iterated Local Search with improvement procedures based on integer programming over a Set Partitioning formulation and a Speed Optimization Algorithm applied only after each local search iteration (ILS-SP-SOA). The obtained results demonstrated that good-quality solutions are reached. The authors pointed out the incorporation of speed decisions in the local search process as a line of future research. However, to the best of our knowledge, no approaches in the literature address the insertion and speed simultaneously, which strongly motivates this work.

In this work, different Variable Neighborhood Search approaches are explored to solve the PRP. This paper is organized as follows. The definition of the problem is presented in Section 2. Section 3 details the variable neighborhood variants and their features. More precisely, we present the method to derive the initial solution and the neighborhood structures. The results and discussion of the computational experiments conducted on benchmark instances are exposed in Section 4. Finally, the main conclusions are drawn in Section 5.

2. PROBLEM DEFINITION

The Pollution-Routing Problem aims to minimize a total cost encompassing fuel, greenhouse gas emissions, and driver costs. This problem consists of determining the routes that allow delivering all the demand of all customers in their time windows, following the speed limits and the maximum capacity of each vehicle, considering that all vehicles used depart from and return to the depot and that each customer is visited exactly once.

The PRP is described as follows. The complete and directed graph $\mathcal{G}=(\mathcal{N},\mathcal{A})$ represents the set of nodes $\mathcal{N}$ and the set of link arcs between them $\mathcal{A}$. The set of nodes is composed of the warehouse (node 0) and the n customers, so $\mathcal{N}=\{0,1,\ldots,n\}$. The set $\mathcal{A}=\{(i,j)\colon i,j \in \mathcal{N} \text{ and } i \neq j\}$ represents the arcs that link each of the different nodes, with the distance between them defined by d_{ij}. There is a homogeneous fleet of m vehicles, each with Q capacity.

Let $\mathcal{N}_0 = \mathcal{N}\backslash\{0\}$ be the set of customers. Each customer $i \in \mathcal{N}_0$ has a demand q_i, a service time required for unloading t_i, and can only be served within its time window $[a_i, b_i]$. In the case of an early arrival at the customer, the driver, and necessarily the vehicle, will have to wait until the moment a_i, while late arrivals (after b_i) are not allowed.

In the PRP, the average speed to traverse each arc (i,j) of the route, $i,j \in \mathcal{N}_0$, is a decision variable v_{ij} that must be within the interval $[v^l, v^u]$. The speed and the total weight of the vehicles strongly influence all the cost components of the problem. Note that the vehicle's weight is the sum of the empty vehicle weight and load to be delivered in a given moment. The first model presented in the literature Bektaş and Laporte [7] uses discretized speed values defined by R speed levels, $\overline{v}^r$ (with $r=1,\ldots,R$). Thus, a binary decision variable z_{ij}^r represents traversing the arc $(i,j) \in \mathcal{A}$ at a default speed at the level r. Binary decision variables x_{ij} indicates whether the arc $(i,j) \in \mathcal{A}$ is traversed (in case $x_{ij}=1$). There are three sets of continuous variables in the model: f_{ij} refer to the total load existing in the vehicle to traverse the arc $(i,j) \in \mathcal{A}$; y_i indicate the instant when node $i \in \mathcal{N}_0$ starts to be served; finally, s_i for any node $i \in \mathcal{N}_0$ represents the last customer to be served before returning to the depot. Note that the variable corresponds to the total route time (which assumes the instant at which node $i \in \mathcal{N}_0$ is served by adding its service t_i, plus the time required to travel the distance d_{i0} at a given speed).

Table 1 presents the parameters used in the objective function of the PRP model.

Table 1: Parameters used in the PRP model.

Notation	Description	Value
f_c	Fuel and CO_2 emissions cost per liter (£)	1.4
k	Engine friction factor (kJ/rev/liter)	0.2
N	Engine speed (rev/second)	33
V	Engine displacement (liters)	5
λ	Expression related to vehicle conditions	0.00003084
f_d	Driver wage per (£/second)	0.0022
γ	Expression related to drive efficiency	0.00277778
α	Expression for vehicle-arc constant	0.0981
β	Expression for vehicle-specific constant	1.64865372

Source: Adapted from Demir et al. [13]

The objective function of the PRP proposed in Demir et al. [13] is as follows:

$$\min f_c \left(\sum_{(i,j)\in\mathcal{A}} kNV\lambda d_{ij} \sum_{r=1}^{R} z_{ij}^r/\overline{v}^r + \sum_{(i,j)\in\mathcal{A}} w\gamma\lambda\alpha d_{ij}x_{ij} + \sum_{(i,j)\in\mathcal{A}} \gamma\lambda\alpha d_{ij}f_{ij} + \sum_{(i,j)\in\mathcal{A}} \beta\gamma\lambda d_{ij} \sum_{r=1}^{R} z_{ij}^r(\overline{v}^r)^2 \right) + \sum_{j\in\mathcal{N}_0} f_d s_j. \tag{1}$$

The first and fourth terms are costs related to the specificities of the engine module and the type of road (varying according to the speed value). The second and third terms are related to the vehicle load on each

arc traveled, considering the vehicle curb weight and payload. The latter term refers to the cost involved with drivers' wages.

3. VARIABLE NEIGHBORHOOD SEARCH APPROACHES

To solve the PRP efficiently, we develop a set of optimization algorithms based on different variants of the Variable Neighborhood Search (VNS) algorithm with innovative neighborhood structures that rely on the particularities of the problem, and we perform parallel and sequential strategies seeking improved solutions in less time. In the sequel, we detail the constructive method of the initial solution (with the particularities and conditions of insertion of a node in an existing route), as well as the neighborhood structures and the different VNS variants.

3.1. Computing an Initial Solution

The initial solution was obtained by the nearest neighbor (customer) algorithm. The constructive method was based on successive insertions of nodes at the end of the routes, in which the average speed of vehicles on the arcs (before and after the inserted node) is equal to v^*_{FD}. The speed value v^*_{FD} is the speed that minimizes fuel and driver wages costs between any arc $(i,j) \in \mathcal{A}$ (Equation 2, Kramer et al. [14]).

$$v^*_{FD} = \left(\frac{\frac{f_c}{f_d} + \lambda k N V}{2\lambda\beta\gamma} \right)^{1/3} \tag{2}$$

Let node $j \in \mathcal{N}_0$ be the node to insert at a given position within a route. Let $i \in \mathcal{N}_0$ be the new predecessor of j and $k \in \mathcal{N}_0$ be the following node. Thus, two new arcs will be added: (i, j) and (j, k), and will be traversed at a speed v_{ij} and v_{jk}, respectively. Additionally, the following conditions are assumed:

- the sum of total load on the vehicle (before the insertion position) with the demand q_j does not exceed the capacity Q of the vehicle;
- when inserting new customers in the routes, the default speeds through the added arcs are equal to v^*_{FD} (v_{ij}=v_{jk}=v^*_{FD}). In an insertion, all customers visited before j keep the same instant they are served. Since the insertion within a route impacts the service time of all subsequent nodes and in the inserted one. Therefore, the new arrival times of customer j and all following customers are recursively verified to ensure they meet their time windows. For instance, let y_j be the minimum time to serve customer $j \in \mathcal{N}_0$ ($y_j = y_i + t_i + d_{ij}/v^*_{FD}$):
  - if the client j is reached within its time window, $a_j \leq y_j \leq b_j$, then it is possible to insert node j in the route;
  - if the vehicle arrives late at the customer j ($y_j > b_j$), traveling the route at a speed v_{ij} will lead to an invalid solution. To tackle this issue, the speed must necessarily be increased. Thus, the speed limit is set at $v^*_{ij} = d_{ij} / (b_j - y_i - t_i)$ so that the arrival time at node j is exactly b_j;
  - if the vehicle arrives early at the customer j, before its time window ($y_j < a_j$), the driver will have to wait until a_j. In this circumstance, although it does not lead to an infeasible solution, the speed could be reduced without compromising the total travel time (the driver will always have to wait until a_j). Therefore, the speed in the arc (i, j) is reduced to the maximum value of the following two expressions:
    - $v^*_{ij} = d_{ij} / (a_j - y_i - t_i)$ to arrive at the customer precisely at a_j;
    - v^*_F which minimizes only fuel costs between any arc $(i,j) \in \mathcal{A}$ in this problem (Equation 3, Demir et al. [13]), as follows:

$$v_F^* = \left(\frac{\lambda k N V}{2\lambda\beta\gamma}\right)^{1/3} \tag{3}$$

- in all situations, speed changes in the arcs must comply with the minimum and maximum speed limits defined by the problem parameters

Solutions that violate time window constraints when inserting a customer could be immediately discarded. However, promising solutions may be lost if the speed over the arcs is not adjusted when inserting a given customer. This approach is particularly innovative: to the best of our knowledge, this is the first work that combines the insertion of a new customer and the speeds on the new arcs. In this way, we aim to incorporate the ability to find other speeds in the arcs as the routes are being formed and not leave this analysis to a post-processing procedure as in Demir et al. [13], which may be important to find good-quality solutions.

3.2. Neighborhood Structures

In all proposed variants, we considered nine different neighborhood structures, which are presented in the sequel and they can be divided into two sets. The set of five structures that exchange the order of visiting customers is as follows:

$\mathcal{N}_1$ – Insert one customer in another route;
$\mathcal{N}_2$ – Exchange between two customers within the same route;
$\mathcal{N}_3$ – Exchange between two customers from different routes;
$\mathcal{N}_4$ – Reverse route;
$\mathcal{N}_5$ – Change one customer in the same route.

Furthermore, another set with four structures that change the speeds on the arcs, where the χ value corresponds to a parameter:

$\mathcal{N}_6$ – Speed down (χ%) all arcs that belong to the one route;
$\mathcal{N}_7$ – Speed up (χ%) all arcs that belong to the one route;
$\mathcal{N}_8$ – Speed down (χ%) one arc of one route;
$\mathcal{N}_9$ – Speed up (χ%) one arc of one route.

All neighborhood structures were used in both phases (shaking and local search) of the VNS algorithms, and only feasible solutions are explored. Typical structures for VRP problems are combined with innovative structures to help with arc speed decisions.

The insertion/removal conditions presented in Section 3.1 were widely applied, mainly in the first set of neighborhood structures that requires customer exchanges.

3.3. Variable Neighborhood Search Algorithms

The addressed VNS algorithms start with an initial feasible solution, built by the greedy heuristic described in Section 3.1. The solution goes through a perturbation phase and, from this, progresses to better solutions through the local search until some stopping condition is reached [15].

The proposed methods are embedded in each other in the sequence that is presented here.

3.3.1. General Variable Neighborhood Search

The General Variable Neighborhood Search (GVNS) is a variant of basic VNS, with a standard shaking phase and its distinguished local search performed through the Variable Neighborhood Descent (VND) method [15]. All nine presented neighborhood structures were considered in the VND phase in their lexicographical order ($\mathcal{N}_{i\in\{1,2,3,4,5,6,7,8,9\}}$).

3.3.2. Smart General Variable Neighborhood Search

The Smart General Variable Neighborhood Search (Smart GVNS) is a more recent method, presented for the first time in Rego and Souza [16].

It specifies that if a better solution is not found in a certain number of iterations, the perturbations in the shaking phase are intensified by applying consecutive perturbations within the neighborhood structure already defined for the shaking phase. If there is any improvement, this intensification stops allowing again to explore solutions less far from the incumbent one.

In Bezerra et al. [17], the authors compared the GVNS approach and the version with "smart" procedures, verifying better solutions in the latter approach, despite the more computational time needed to obtain the results. This adaptability of the algorithm to reach solutions is essential to avoid traps and, at the same time, not lead to significant disturbances that arise from very distant solutions (taking more time to converge to the new local optimum) [18].

3.3.3. Smart and Parallel General Variable Neighborhood Search

The application of parallelization strategies is a promising line of research for large-size problems. Computational capabilities are increasing and reaching better results in the same computational time spent in a sequential approach or, in other words, to obtain the same results in a shorter computational time [19]. Parallelization strategies were associated with VNS for the first time in García-López et al. [20], presenting approaches that sought to achieve more efficiency by increasing the exploration of the solution space compared to a sequential approach.

The implementation of the Smart and Parallel General Variable Neighborhood Search approach (Smart and Parallel GVNS) followed the principles of the smart version referred to in the previous subsection with the parallelization strategy Replicated Shaking VNS: starting from the incumbent solution, shaking and local search (VND) are executed in parallel.

For the sake of clarity, the Smart and Parallel GVNS algorithm to solve the PRP is summarized in Figure 1. Starting from an initial solution, until the stopping criterion is not reached and for each neighborhood structure (k), n_{max} blocks are executed in parallel. Each block comprises the perturbation phase (with an intensity of p consecutive perturbations) and VND phase. At the end of the parallel blocks, if the best solution found is worse than the incumbent, the value of p is increased. It is assumed that it is enough not to improve in 1 iteration to intensify the shaking phase. Otherwise, a new iteration is initialized with a new incumbent solution, neighborhood structure k=1, and p returning to a value of 1, allowing again to explore solutions less distant from the incumbent solution. The neighborhood structure to be explored is modified only when p_{max} is exceeded (value of k is incremented).

The Smart GVNS approach can be derived by setting the maximum number of blocks as n_{max}=1 (i.e., no processes are running simultaneously). The GVNS can also be derived, assuming n_{max}=1 and p_{max}=0.

4. COMPUTATIONAL EXPERIMENTS

All the proposed approaches were evaluated through extensive computational experiments on benchmark instances proposed in the literature (real instances from the United Kingdom (UK) case study). In Demir et al. [13], different sets of instances based on real geographic distances between cities from the UK were compiled into a library of PRP (PRPLIB). Each instance is represented by Kn_o, where o is the ordinal number of the instance for n customers.

All the algorithms were coded in Java version 8, and all experiments were conducted on a PC with an Intel 8th Gen Core i7-8565 processor, 16 GB of RAM.

As referred to above, nine neighborhood structures were used (k_{max}=9). The percentage variation of speeds in the neighborhood structures $\mathcal{N}_{i\in\{6,7,8,9\}}$ were set at χ=0.10. The maximum number of parallel blocks was n_{max}=10, and the maximum number of consecutive movements in the same neighborhood structure was p_{max}=5.

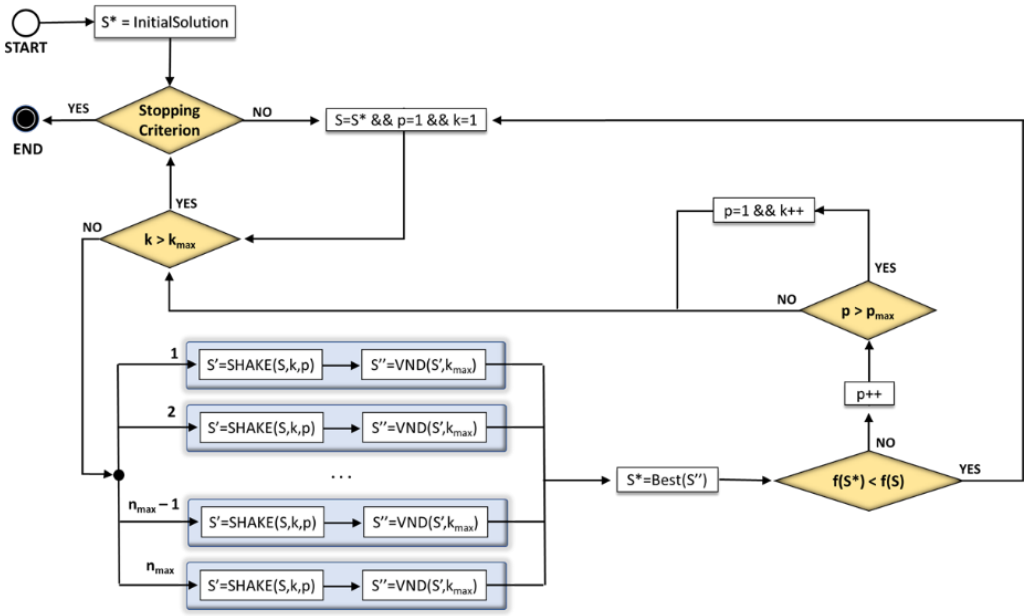

Figure 1: Flowchart of Smart and Parallel GVNS algorithm

The stopping criterion for all approaches was set at t_{max}=5 seconds, but without forcing its end (i.e., if better solutions are obtained in the internal cycles of the algorithm, its execution continues). Therefore, solutions can be obtained well after 5 seconds.

Given the stochastic component of the proposed algorithms, ten runs were conducted for each instance and approach. We compare the best results obtained by GVNS, Smart GVNS (S_GVNS), and Smart and Parallel GVNS (SP_GVNS) approaches between them and with the results obtained in Demir et al. [13] (ALNS) and Kramer et al. [14] (ILS-SP-SOA) for instances with 10, 50, and 100 customers. We compare our best global results with those existing in the literature (comparing the gap with other approaches). The results are summarized in Tables 2-4, and the meaning of each column is the following:

- *time*: average time to obtain the best solution;
- *cost*: value of the minimum cost solution;
- gap_{ALNS}: average gap (in %) among our lower cost solution with the one obtained in Demir et al. [13], as follows: ((our best cost - ALNS cost) / ALNS cost);
- $gap_{ILS-SP-SOA}$: average gap (in %) among our lower cost solution with the one obtained in Kramer et al. [14], as follows: ((our best cost - ILS-SP-SOA cost) / ILS-SP-SOA cost).

The best solution from the three VNS approaches appears in bold. As can be seen, the SP_GVNS approach outperforms the others for any of the sets of instances. Between GVNS and S_GVNS, the latter presents better results, except for the small-size instances in which GVNS reaches almost the same results in less computational time.

In general, and comparing the best solutions with those in the literature, the VNS approaches tend to achieve better solutions than in Demir et al. [13] for all sets of instances. The gap_{ALNS} values indicate that our best solutions have, on average, less -0.41%, -0.05%, and -0.57% of the costs than those obtained through ALNS, for the set of instances with 10, 50, and 100 customers, respectively. Furthermore, our solutions were obtained in less or a similar computational time.

Compared to the ILS-SP-SOA approach Kramer et al. [14], $gap_{ILS-SP-SOA}$ suggests that our approaches obtained a lower cost of -0.39% (on average) in the smallest-size instances (10 customers). For 50- and 100-customer instances, the cost increased 0.46% and 1.26%, respectively. The proposed VNS approaches tend

Table 2: Computational results for PRP instances with 10 customers.

Instance	GVNS		S_GVNS		SP_GVNS		ALNS		ILS-SP-SOA		OUR BEST	
	time	cost	time	cost	time	cost	time	cost	time	cost	gap_{ALNS}	$gap_{ILS-SP-SOA}$
UK10_01	0.08	169.96	0.02	169.96	0.02	**169.96**	2.1	170.64	0.04	170.64	-0.40%	-0.40%
UK10_02	0.10	204.12	0.03	**204.12**	0.17	204.12	2.3	204.88	0.05	204.88	-0.37%	-0.37%
UK10_03	0.06	**199.54**	0.09	199.54	0.02	199.54	2	200.42	0.03	200.4	-0.44%	-0.43%
UK10_04	0.06	189.14	0.02	**189.14**	0.05	189.14	2.2	189.99	0.04	189.88	-0.45%	-0.39%
UK10_05	0.10	174.91	0.09	174.91	0.09	**174.91**	2.3	175.59	0.04	175.59	-0.39%	-0.39%
UK10_06	0.05	**220.91**	0.07	220.91	0.30	213.77	2.2	214.48	0.05	214.48	-0.33%	-0.33%
UK10_07	0.01	189.44	0.01	**189.44**	0.01	189.44	2.9	190.14	0.04	190.14	-0.37%	-0.37%
UK10_08	0.05	221.33	0.04	221.33	0.03	**221.33**	2.1	222.17	0.03	222.17	-0.38%	-0.38%
UK10_09	0.02	173.89	0.04	173.89	0.01	**173.89**	2.2	174.54	0.04	174.54	-0.37%	-0.37%
UK10_10	0.12	189.12	0.12	189.12	0.17	**189.12**	2.6	190.04	0.04	189.82	-0.49%	-0.37%
UK10_11	0.06	261.14	0.06	261.14	0.06	**261.14**	2.2	262.08	0.03	262.08	-0.36%	-0.36%
UK10_12	0.02	182.47	0.01	182.47	0.02	**182.47**	2.2	183.19	0.04	183.19	-0.39%	-0.39%
UK10_13	0.01	195.26	0.03	195.26	0.00	**195.26**	2.2	195.97	0.04	195.97	-0.36%	-0.36%
UK10_14	0.05	162.50	0.30	162.50	0.09	**162.50**	2.4	163.28	0.04	163.17	-0.48%	-0.41%
UK10_15	0.03	126.65	0.04	126.65	1.06	**126.55**	2.4	127.24	0.05	127.1	-0.54%	-0.43%
UK10_16	0.03	185.86	0.02	**185.86**	0.03	185.86	1.9	186.73	0.04	186.63	-0.47%	-0.41%
UK10_17	0.06	158.39	0.05	158.39	0.03	**158.39**	2.3	159.03	0.04	159.03	-0.40%	-0.40%
UK10_18	0.04	161.46	0.02	**161.46**	0.21	161.46	2.2	162.09	0.04	162.09	-0.39%	-0.39%
UK10_19	0.04	168.82	0.08	**168.79**	0.13	168.79	4.1	169.59	0.04	169.46	-0.47%	-0.40%
UK10_20	0.03	168.12	0.05	168.12	0.01	**168.12**	2	168.8	0.03	168.8	-0.40%	-0.40%
avg.	0.05	185.15	0.06	185.15	0.13	184.79	2.34	185.5445	0.04	185.503	-0.41%	-0.39%

Table 3: Computational results for PRP instances with 50 customers.

Instance	GVNS		S_GVNS		SP_GVNS		ALNS		ILS-SP-SOA		OUR BEST	
	time	cost	time	cost	time	cost	time	cost	time	cost	gap_{ALNS}	$gap_{ILS-SP-SOA}$
UK50_01	4.87	638.56	4.26	594.29	11.03	**590.64**	29.7	593.77	2.58	593.14	-0.53%	-0.42%
UK50_02	4.12	615.14	3.94	609.62	8.68	**599.49**	60.3	599.43	2.61	599.66	0.01%	-0.03%
UK50_03	2.81	629.32	4.09	640.11	12.04	**619.74**	53.4	626.21	2.77	621.66	-1.03%	-0.31%
UK50_04	1.45	768.12	3.64	765.87	4.80	**735.18**	35.8	740.92	2.97	738.18	-0.77%	-0.41%
UK50_05	3.94	692.21	4.49	656.21	14.06	**643.83**	35.3	636	2.41	632.77	1.23%	1.75%
UK50_06	4.68	625.71	5.31	584.84	12.27	**581.86**	54.6	584.61	2.92	584.47	-0.47%	-0.45%
UK50_07	4.97	554.93	4.17	544.22	16.14	**539.28**	25.7	541.07	3.43	536.98	-0.33%	0.43%
UK50_08	2.19	635.16	4.12	567.81	17.45	**561.27**	39.5	560.27	2.79	558.66	0.18%	0.47%
UK50_09	4.26	708.24	4.43	686.11	8.99	**681.57**	21.4	687.79	2.8	683.38	-0.90%	-0.26%
UK50_10	4.99	726.64	4.69	706.70	13.81	**681.72**	25.6	670.92	5.03	664.58	1.61%	2.58%
UK50_11	4.39	647.40	2.94	642.47	4.98	**620.68**	25.1	618.94	3.21	618.29	0.28%	0.39%
UK50_12	4.17	572.88	3.79	571.64	16.02	**569.72**	40.7	571.42	2.37	570.72	-0.30%	-0.17%
UK50_13	4.84	643.05	3.92	587.11	10.63	**584.08**	52.1	589.11	2.32	586.68	-0.85%	-0.44%
UK50_14	5.02	711.82	4.23	683.50	6.79	**672.38**	35	660.17	3.62	655.9	1.85%	2.51%
UK50_15	4.70	614.56	4.40	598.76	8.43	**601.94**	26.2	584.13	2.74	584.24	2.50%	2.49%
UK50_16	5.02	600.46	4.24	585.50	9.90	**577.38**	51	585.16	2.9	574.63	-1.33%	0.48%
UK50_17	4.95	480.42	4.69	467.79	15.86	**455.71**	20	456.56	2.45	456.61	-0.19%	-0.20%
UK50_18	4.30	714.42	4.07	697.47	5.88	**683.79**	26.4	681.72	3.4	679.95	0.30%	0.56%
UK50_19	4.00	624.43	3.78	598.84	6.56	**591.01**	21.2	597.95	3.45	588.17	-1.16%	0.48%
UK50_20	5.14	681.26	4.53	683.62	9.21	**670.61**	28.9	678.56	2.8	672.21	-1.17%	-0.24%
avg.	4.24	644.24	4.19	623.62	10.67	613.09	35.395	613.2355	2.98	610.044	-0.05%	0.46%

to spend more computational time than the ILS-SP-SOA approach. However, it is important to emphasize that competitive and good-quality solutions are obtained in less time than the ILS-SP-SOA approach for the smallest-size instances. The higher computational time shown in Table 2 is due to the fact that any subsequent improvement dictates a new VNS iteration and thus an increasing the average computational time to obtain final solutions.

5. CONCLUSIONS

In this paper, we propose a set of new optimization approaches to the Pollution-Routing Problem. More precisely, we developed algorithms based on different variants of the Variable Neighborhood Search algorithm (VNS). Furthermore, parallel and sequential strategies were applied and tested in benchmark instances. Our approaches were compared between them and with state-of-the-art solutions.

Among the proposed approaches (GVNS, Smart GVNS and Smart and Parallel GVNS), Smart and Parallel variant presents better results, but it can take more time to reach the final solution. Due to innovative neigh-

Table 4: Computational results for PRP instances with 100 customers.

Instance	GVNS		S_GVNS		SP_GVNS		ALNS		ILS-SP-SOA		OUR BEST	
	time	cost	time	cost	time	cost	time	cost	time	cost	gap_{ALNS}	$gap_{ILS-SP-SOA}$
UK100_01	4.69	1293.46	7.90	1270.59	122.35	**1227.14**	92.1	1240.79	34.65	1210.63	-1.10%	1.36%
UK100_02	4.92	1236.58	10.85	1190.19	137.58	**1147.91**	98.2	1168.17	33.2	1149.07	-1.73%	-0.10%
UK100_03	4.82	1144.70	13.25	1117.85	212.56	**1089.46**	207.9	1092.73	33.28	1079.53	-0.30%	0.92%
UK100_04	4.68	1143.51	8.72	1116.08	141.79	**1091.60**	149.7	1106.48	35.79	1076.46	-1.34%	1.41%
UK100_05	4.92	1094.08	7.80	1092.57	175.65	**1053.25**	159	1043.41	33.63	1032.64	0.94%	2.00%
UK100_06	4.66	1303.94	8.22	1284.20	107.01	**1202.50**	133.8	1213.61	29.37	1193.86	-0.92%	0.72%
UK100_07	4.73	1172.24	7.19	1141.95	193.83	**1056.40**	102.6	1060.08	28.63	1046.92	-0.35%	0.91%
UK100_08	4.76	1182.45	6.54	1170.97	124.93	**1109.24**	209.5	1106.78	26.63	1091.27	0.22%	1.65%
UK100_09	4.79	1080.42	8.69	1066.36	191.45	**1004.15**	154	1015.46	30.47	989.66	-1.11%	1.46%
UK100_10	4.50	1161.49	8.63	1132.75	157.92	**1078.33**	199	1076.56	29.73	1061.42	0.16%	1.59%
UK100_11	4.63	1346.85	14.03	1238.64	209.09	**1218.34**	107.1	1210.25	36.26	1198.08	0.67%	1.69%
UK100_12	4.81	1095.24	10.12	1065.43	144.15	**1039.20**	206.4	1053.02	31.91	1028.95	-1.31%	1.00%
UK100_13	4.60	1227.33	7.24	1193.77	110.91	**1147.75**	87.9	1154.83	27.46	1132.72	-0.61%	1.33%
UK100_14	4.81	1343.49	7.57	1358.03	98.57	**1276.56**	91.8	1264.5	31.45	1242.68	0.95%	2.73%
UK100_15	4.59	1386.91	9.15	1344.03	142.16	**1300.10**	110.9	1315.5	36.24	1302.19	-1.17%	-0.16%
UK100_16	4.57	1062.27	10.61	1047.29	173.87	**994.94**	254.7	1005.03	28.14	982.77	-1.00%	1.24%
UK100_17	5.05	1363.89	14.72	1316.34	184.70	**1271.99**	152.8	1284.81	38.88	1259.07	-1.00%	1.03%
UK100_18	4.48	1177.21	8.49	1135.40	151.53	**1099.48**	92.6	1106	33.15	1079.79	-0.59%	1.82%
UK100_19	4.37	1093.34	13.19	1068.20	221.21	**1032.37**	91	1044.71	30.67	1017.22	-1.18%	1.49%
UK100_20	4.79	1374.15	10.58	1314.92	151.16	**1255.05**	204.4	1263.06	30.05	1241.72	-0.63%	1.07%
avg.	4.71	1214.18	9.68	1183.28	157.62	1134.79	145.27	1141.289	31.98	1120.833	-0.57%	1.26%

borhood structures that rely on the particularities of this PRP, in which the vehicle load and speed significantly influence the total cost in each arc, one was able to search for new solution spaces that led to better solutions than those existing in the literature (particularly for the small-size instances). These better solutions were due to the ability to adjust the speed values in the arcs in an integrated and direct way in the existing routes, allowing new promising solutions.

Acknowledgements

The first author has been supported by FCT – Fundação para a Ciência e Tecnologia, through national funds from MCTES – Ministério da Ciência, Tecnologia e Ensino Superior, and by European Social Fund through NORTE2020 – Programa Operacional Regional Norte, within the research grant SFRH/BD/146217/2019. This work has been supported by FCT – Fundação para a Ciência e a Tecnologia within the R&D Units Project Scope: UIDB/00319/2020.

References

[1] Y. Suzuki, A dual-objective metaheuristic approach to solve practical pollution routing problem, International Journal of Production Economics 176 (2016) 143–153. doi:10.1016/J.IJPE.2016.03.008.

[2] E. Ericsson, H. Larsson, K. Brundell-Freij, Optimizing route choice for lowest fuel consumption - Potential effects of a new driver support tool, Transportation Research Part C: Emerging Technologies 14 (2006) 369–383. doi:10.1016/j.trc.2006.10.001.

[3] A. Tiwari, P. C. Chang, A block recombination approach to solve green vehicle routing problem, International Journal of Production Economics 164 (2015) 379–387. doi:10.1016/J.IJPE.2014.11.003.

[4] A. Luè, C. Bresciani, A. Colorni, F. Lia, V. Maras, Z. Radmilović, L. Whitmarsh, D. Xenias, E. Anoyrkati, Future priorities for a climate-friendly transport: A European strategic research agenda toward 2030, International Journal of Sustainable Transportation 10 (2016) 236–246. doi:10.1080/15568318.2014.893043.

[5] R. Fukasawa, Q. He, Y. Song, A disjunctive convex programming approach to the pollution-routing problem, Transportation Research Part B: Methodological 94 (2016) 61–79. doi:10.1016/J.TRB.2016.09.006.

[6] E. Demir, T. Bektaş, G. Laporte, A review of recent research on green road freight transportation 237 (2014) 775–793. doi:10.1016/j.ejor.2013.12.033.

[7] T. Bektaş, G. Laporte, The Pollution-Routing Problem, Transportation Research Part B: Methodological 45 (2011) 1232–1250. doi:10.1016/j.trb.2011.02.004.

[8] M. Barth, T. Younglove, G. Scora, UC Berkeley Research Reports Title Development of a Heavy-Duty Diesel Modal Emissions and Fuel Consumption Model Permalink https://escholarship.org/uc/item/67f0v3zf Publication Date, 2005.

[9] M. Barth, K. Boriboonsomsin, Energy and emissions impacts of a freeway-based dynamic eco-driving system, Transportation Research Part D: Transport and Environment 14 (2009) 400–410. doi:10.1016/j.trd.2009.01.004.

[10] EEA, Trends and projections in Europe 2022, EEA Report No 10/2022, European Environment Agency (https://www.eea.europa.eu/publications/trends-and-projections-in-europe-2022) accessed 28 June 2023., 2022.

[11] S. Dabia, E. Demir, T. Van Woensel, An exact approach for a variant of the pollution-routing problem, Transportation Science 51 (2017) 607–628. doi:10.1287/TRSC.2015.0651.

[12] R. Fukasawa, Q. He, F. Santos, Y. Song, A joint vehicle routing and speed optimization problem, INFORMS Journal on Computing 30 (2018) 694–709. doi:10.1287/ijoc.2018.0810. arXiv:1602.08508.

[13] E. Demir, T. Bektaş, G. Laporte, An adaptive large neighborhood search heuristic for the Pollution-Routing Problem, European Journal of Operational Research 223 (2012) 346–359. doi:10.1016/j.ejor.2012.06.044.

[14] R. Kramer, A. Subramanian, T. Vidal, L. F. dos Anjos Cabral, A matheuristic approach for the Pollution-Routing Problem, European Journal of Operational Research 243 (2015) 523–539. doi:10.1016/j.ejor.2014.12.009.

[15] P. Hansen, N. Mladenović, R. Todosijević, S. Hanafi, Variable neighborhood search: basics and variants, EURO Journal on Computational Optimization 5 (2017) 423–454. doi:10.1007/s13675-016-0075-x.

[16] M. F. Rego, M. J. F. Souza, Smart general variable neighborhood search with local search based on mathematical programming for solving the unrelated parallel machine scheduling problem, in: ICEIS 2019 - Proceedings of the 21st International Conference on Enterprise Information Systems, volume 1, 2019, pp. 275–283. doi:10.5220/0007703302870295.

[17] S. N. Bezerra, M. J. F. Souza, S. R. de Souza, A variable neighborhood search-based algorithm with adaptive local search for the Vehicle Routing Problem with Time Windows and multi-depots aiming for vehicle fleet reduction, Computers & Operations Research 149 (2023) 106016. doi:10.1016/J.COR.2022.106016.

[18] P. Karakostas, A. Sifaleras, A Double-Adaptive General Variable Neighborhood Search algorithm for the solution of the Traveling Salesman Problem, Applied Soft Computing 121 (2022) 108746. doi:10.1016/j.asoc.2022.108746.

[19] J. Brimberg, S. Salhi, R. Todosijević, D. Urošević, Variable Neighborhood Search: The power of change and simplicity, Computers & Operations Research 155 (2023) 106221. doi:10.1016/J.COR.2023.106221.

[20] F. García-López, B. Melián-Batista, J. A. Moreno-Pérez, J. M. Moreno-Vega, The parallel variable neighborhood search for the p-median problem, Journal of Heuristics 8 (2002) 375–388. doi:10.1023/A:1015013919497.