UNIVERSIDADE FEDERAL DE MINAS GERAIS
Engineering Department
Graduate Program in Electrical Engineering

Babak Rezaei

Combining Genetic Local Search into Multi-Population Evolutionary Algorithms for the Capacitated Vehicle Routing Problem

Belo Horizonte

2023

Babak Rezaei

Combining Genetic Local Search into Multi-Population Evolutionary Algorithms for the Capacitated Vehicle Routing Problem

Thesis presented to the Graduate Program in Electrical Engineering at the Universidade Federal de Minas Gerais, as a partial requirement for obtaining the title of Doctor in Electrical Engineering.

Supervisor:
Prof. Dr. Frederico Gadelha Guimarães
Co-supervisors:
Prof. Dr. Pauline Catriona Haddow
Prof. Dr. Rasul Enayatifar

Belo Horizonte
2023

Rezaei, Babak.
R467c Combining genetic local search into multi-population evolutionary algorithms for the capacitated vehicle routing problem [recurso eletrônico] / Babak Rezaei. - 2023.
1 recurso online (109 f. : il., color.) : pdf.

Orientador: Frederico Gadelha Guimarães.
Coorientadores: Pauline Catriona Haddow, Rasul Enayatifar.

Tese (doutorado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Bibliografia: f. 100-109.
Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia elétrica - Teses. 2. Veículos - Teses 3. Algoritmos genéticos - Teses. 4. Genética - Teses. 5. Algoritmos - Teses. 6. Inteligência artificial - Teses. 7. Ciência da computação - Teses. 8. Cálculos numéricos - Teses. I. Guimarães, Frederico Gadelha. II. Haddow, Pauline Catriona. III. Enayatifar, Rasul. IV. Universidade Federal de Minas Gerais. Escola de Engenharia. V. Título.

CDU: 621.3(043)

Ficha catalográfica elaborada pela bibliotecária Ângela Cristina Silva CRB/6 2361
Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

FOLHA DE APROVAÇÃO

"COMBINING GENETIC LOCAL SEARCH INTO MULTI-POPULATION EVOLUTIONARY ALGORITHMS FOR THE CAPACITATED VEHICLE ROUTING PROBLEM"

BABAK REZAEI

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica. Aprovada em 30 de agosto de 2023. Por:

Prof. Dr. Frederico Gadelha Guimarães - Orientador
DCC (UFMG)

Prof. Dr. Pauline Catriona Haddow - Coorientadora
Department of Computer Science (Norwegian University of Science and Technology)

Prof. Dr. Rasul Enyatifar - Coorientador
Department of Computer Engineering (Islamic Azad University)

Prof. Dr. Gilberto Reynoso Meza
(PUC-PR)

Prof. Dr. Puca Huachi Vaz Penna
DECOM (UFOP)

Prof. Dr. Roberto Gomes Ribeiro
DECSI (UFOP)

Prof. Dr. Lucas de Souza Batista
DEE (UFMG)

Documento assinado eletronicamente por **Frederico Gadelha Guimaraes**, **Coordenador(a) de curso de pós-graduação**, em 01/09/2023, às 14:28, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2582207** e o código CRC **C0F4E6DA**.

Referência: Processo nº 23072.252507/2023-01 SEI nº 2582207

To my beloved family,

To my wife and our wonderful children,

This achievement is not mine alone; it belongs to all of us. Thank you for standing by my side, for believing in me when I doubted myself, and for being the source of my inspiration. You are my rock, and I dedicate this thesis to you with all my love.

With deepest gratitude,

Babak

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisors, Prof. Dr. Frederico Gadelha Guimarães, Prof. Dr. Rasul Enayatifar and Prof. Dr. Pauline Catriona Haddow, for their unwavering support, guidance, and invaluable mentorship throughout the course of my research. I am also deeply thankful to the members of my thesis committee, Prof. Dr. Gilberto Reynoso Meza, Prof. Dr. Puca Huachi Vaz Penna, Prof. Dr. Roberto Gomes Ribeiro and Prof. Dr. Lucas de Souza Batista, for their insightful feedback and constructive criticism that greatly improved the quality of this thesis.

I extend my heartfelt thanks to my family for their unwavering support, encouragement, and understanding during this challenging journey. Your love and belief in me sustained me through the toughest times.

My appreciation goes to my lab mates at MINDS, and colleagues at Concert Technologies, whose camaraderie and intellectual exchange enriched my experience.

Lastly, I want to thank all the individuals, too numerous to name individually, who contributed in various ways to my research and academic growth.

This thesis would not have been possible without the collective support and encouragement of all those mentioned above. Your contributions have been invaluable, and I am deeply grateful for your presence in my academic journey.

RESUMO

O Problema de Roteamento de Veículos (VRP) é um dos problemas mais significativos na pesquisa operacional atualmente. O VRP tem uma ampla gama de campos de aplicação, como transporte, logística, manufatura, sistemas de auxílio e comunicação. Para atender às necessidades de diferentes cenários do VRP no mundo real, muitos modelos de VRP foram desenvolvidos - sendo o CVRP (VRP capacitado) a forma clássica. Neste estudo, é proposto inicialmente um algoritmo híbrido (ICAHGS) para resolver o CVRP, combinando um ICA (Algoritmo Competitivo Imperialista) refinado como o método evolucionário primário e de múltiplas populações, e um algoritmo de Busca Genética Híbrida (HGS-CVRP) como uma estratégia aprimorada de busca local e gerenciamento de população dentro do framework do ICA. O ICAHGS foi comparado com diversos algoritmos de ponta da literatura. Os resultados dessa comparação, que incluem tanto instâncias de referência clássicas quanto aplicações do mundo real, demonstram o desempenho competitivo do algoritmo proposto. Posteriormente, é introduzido o Algoritmo Genético de Ilhas com População Dinâmica e HGS (DPIGA-HGS), que é um novo modelo híbrido de metaheurística. O DPIGA-HGS integra um modelo de ilhas especializado (DPIGA) e um HGS refinado como seu mecanismo de busca local dentro de cada ilha. O objetivo principal do DPIGA-HGS é contribuir para o avanço do campo, propondo uma nova variante do Algoritmo Genético de Ilhas e, simultaneamente, alcançando resultados de otimização aprimorados em comparação com o ICAHGS. Os resultados das análises comparativas revelaram o desempenho superior do DPIGA-HGS quando comparado a outros algoritmos de ponta, incluindo o ICAHGS. Através de múltiplos conjuntos de dados de referência, o DPIGA-HGS demonstrou sua habilidade ao alcançar um número significativo de solução mais conhecida (BKS), superando seus concorrentes em várias instâncias.

Palavras-chave: Problema de Roteamento de Veículos, Computação Evolutiva, Algoritmo Competitivo Imperialista, Pesquisa Genética Híbrida, Algoritmo Genético de Ilha, Algoritmo Genético Multi-populacional.

ABSTRACT

The Vehicle Routing Problem (VRP) is one of the most significant problems in operational research today. VRP has a vast range of application fields such as transportation, logistics, manufacturing, relief systems and communication. To suit the needs of different real-world VRP scenarios, many models of VRP have been developed - CVRP (Capacitated VRP) being the classical form. In this study, at first a hybrid algorithm (ICAHGS) for solving CVRP is proposed, combining a refined ICA (Imperialist Competitive Algorithm) as the primary evolutionary and multi-population method, and a Hybrid Genetic Search (HGS-CVRP) algorithm as an enhanced local search and population management strategy within the ICA framework. ICAHGS has been compared to several state-of-the-art algorithms from literature. The results of this comparison, which include both classical benchmark instances and real-world applications, demonstrate the competitive performance of the proposed algorithm. Afterwards, Dynamic Population Island GA and HGS (DPIGA-HGS) is introduced, which is a novel hybrid metaheuristic model. DPIGA-HGS integrates a specialized island model (DPIGA) and a refined HGS as its local search engine within each island. The primary objective of DPIGA-HGS is to contribute to the advancement of the field by proposing a new variant of Island GA and simultaneously achieving improved optimization results in comparison to ICAHGS. The results of the comparative analyses revealed the superior performance of DPIGA-HGS when pitted against other state-of-the-art algorithms, including ICAHGS. Across multiple benchmark datasets, DPIGA-HGS showcased its prowess by achieving a significant number of BKS (Best Known Solution), outperforming its competitors in various instances.

Keywords: Vehicle Routing Problem, Evolutionary Computation, Imperialist Competitive Algorithm, Hybrid Genetic Search, Island Genetic Algorithm, Multi-Population Genetic Algorithm.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS AND ACRONYMS

ACO	Ant Colony Optimization
CARP	Capacitated Arc Routing Problem
CVRP	Capacitated Vehicle Routing Problem
EA	Evolutionary Algorithm
GA	Genetic Algorithm
GRASP	Greedy Randomize Adaptive Search Procedure
ICA	Imperialist Competition Algorithm
IGA	Island Genetic Algorithm
ILS	Iterated Local Search
LNS	Large Neighborhood Search
LRP	Location Routing problem
MA	Memetic Algorithm
MIP	Mixed Integer Programming
NN	Nearest Neighborhood
OR	Operational Research
PR	Path Relinking
PSO	Particle Swarm Optimizations
SA	Simulated Annealing
SS	Scatter Search
TS	Tabu Search
TSP	Traveling Salesman Problem
VND	Variable Neighborhood Descent
VNS	Variable Neighborhood Search
VRP	Vehicle Routing Problem

CONTENTS

1 INTRODUCTION

1.1 Context and Motivation

The rapid increase in online shopping in our daily lives, requiring a large number of package deliveries, has increased the demand for express transportation and a reduction of transportation costs. Similar demands are placed on transportation systems for emergency healthcare deliveries. Efficient routing is needed to meet such demands, by improving transport allocation and scheduling. Thus, one of the most important problems and success stories of operational research to date, is the vehicle routing problem (VRP), first introduced by Dantzig and Ramser in 1959 (DANTZIG; RAMSER, 1959).

The Vehicle Routing Problem (VRP) is a combinatorial optimization problem that aims to find the most efficient routes for a fleet of vehicles to deliver goods to geographically dispersed customers. VRP has a broad range of applications in logistics, transportation, and supply chain management. In logistics, it is used to optimize delivery routes from a warehouse or distribution center to customers, taking into account factors such as vehicle capacity, road network constraints, and delivery time windows. In transportation, it is applied to optimize the routes of buses, taxis, and other public transportation vehicles to minimize travel time and costs. VRP is also relevant to the design of efficient and cost-effective supply chain networks, determining the most cost-effective mode of transportation, routing and inventory management decisions. Additionally, it can be applied to disaster relief operations and to optimize the routes of mobile sales or service teams (TOTH; VIGO, 2014).

1.2 Aim and Objectives

The primary objective of this research study is to develop a bio-inspired metaheuristic algorithm that efficiently solves the Capacitated Vehicle Routing Problem (CVRP). The performance of the algorithm will be evaluated by conducting experiments on several standardized benchmarks. To fulfill the aim of this study, the following objectives have been identified:

1. **Algorithmic advancement:** Develop and evaluate advanced metaheuristic algorithms for solving combinatorial optimization problems, with a primary focus on enhancing the performance and efficiency of CVRP solutions.

2. **Performance assessment:** Assess the effectiveness of the proposed algorithms in terms of their ability to find high-quality solutions within reasonable computational

timeframes, considering both solution quality and efficiency.

3. **Algorithm comparison:** Conduct a comparative analysis of the proposed algorithms against existing state-of-the-art methods to establish their competitiveness and identify potential areas of improvement.

4. **Algorithm applicability:** Investigate the broader applicability of the developed algorithms to address a range of vehicle routing and logistics optimization challenges, beyond CVRP, thereby contributing to the field of operational research and transportation logistics.

1.3 Main contributions

The main contributions of this study include:

- Introducing two novel hybrid algorithms specifically designed for addressing the CVRP, incorporating innovative strategies and techniques.
- introducing a novel variant of the island model, tailored to the specific requirements and characteristics of the problem domain, thereby contributing to the advancement of the field.

1.4 Research Questions

To accomplish the objectives of this research, the following research questions should be answered:

1. How can an evolutionary algorithm be effectively integrated with the HGS local search to enhance the solution quality and convergence speed for the CVRP?
2. How does the hybrid method compare to other state-of-the-art metaheuristics and hybrid algorithms for the CVRP?
3. What are the advantages of incorporating a multi-population model in the hybrid method for the CVRP compared to single-population approaches?
4. How does the hybrid method balance diversification and intensification strategies to explore the solution space effectively and escape local optima in the CVRP?

1.5 Thesis Outline

- Chapter 2: Background
  This chapter presents the necessary background and fundamental concepts required for understanding the research. It provides an overview of the topic and establishes the foundation for the subsequent chapters.

- Chapter 3: Literature Review
  The literature review chapter offers a comprehensive analysis of previous studies and research conducted by other authors in the field. It discusses the existing literature, identifies research gaps, and highlights relevant findings and methodologies.

- Chapter 4: ICAHGS approach
  In this chapter, the first proposed hybrid model, ICAHGS, is introduced. The chapter elaborates on the model's design, including its components and algorithmic details. The experimental results obtained from applying ICAHGS to different benchmarks are presented and thoroughly analyzed.

- Chapter 5: DPIGA-HGS approach
  This chapter presents the second proposed hybrid model, DPIGA-HGS. It provides a detailed explanation of the model's architecture, incorporating the specialized island model and the HGS as its local search component. The experimental results obtained using DPIGA-HGS are presented and analyzed.

- Chapter 6: Discussion and Conclusion
  The final chapter concludes the thesis by summarizing the key findings, contributions, and implications of the research. It also discusses potential areas for future exploration and suggests possible avenues for further improvement and development of the proposed hybrid models.

2 BACKGROUND

In this chapter, the background work of this research and the basic information that is required for this study will be discussed.

2.1 Vehicle Routing Problem (VRP)

2.1.1 What is VRP?

The Vehicle Routing Problem (VRP) was initially introduced by Dantzig and Ramser in 1959 (DANTZIG; RAMSER, 1959). Their work presented a matching-based heuristic for routing gasoline delivery trucks. Since its inception, the VRP has garnered significant attention in the field of Operational Research (OR) due to its wide range of practical applications in transportation, logistics, distribution, manufacturing, and other domains.

The Vehicle Routing Problem (VRP) can be viewed as an extension or generalization of the well-known Traveling Salesman Problem (TSP) (ROBINSON, 1949), which is a classic problem in the field of operational research. The TSP aims to determine the shortest route that visits all given points exactly once and returns to the starting point. On the other hand, the VRP involves assigning routes to a fleet of vehicles, where each vehicle must visit a specific set of nodes while minimizing the total cost of the operation. Unlike the TSP, the VRP incorporates additional constraints and complexities related to vehicle capacities, customer demands, and vehicle routing logistics.

2.1.2 Types of VRP variants

The original form of the VRP, known as the Capacitated VRP (*CVRP*), aims to determine a set of routes for a fleet of identical vehicles in such a way that all customers are served while minimizing the overall route cost. Each customer must be visited exactly once, and only one route is assigned to each vehicle. Additionally, all vehicles start and finish their routes at a central depot. The CVRP does not allow for splitting deliveries among multiple vehicles. Each vehicle's loading (capacity) and traveling distance must not exceed their respective maximum allowable limits (AKPINAR, 2016).

To better align with real-life scenarios, additional constraints have been incorporated into the original VRP, leading to numerous variants. While not exhaustive, the following classes highlight several of these variations. Figure 1 illustrates the hierarchy of VRP variants.

1. The Multi-depot Vehicle Routing Problem (MDVRP) is a variant of the original VRP where multiple depots are available for vehicles to start and finish their routes. Unlike the standard VRP where there is a single central depot, the MDVRP allows for multiple depots from which vehicles can serve customers (MONTOYA-TORRES et al., 2015).

2. The Pickup and Delivery Vehicle Routing Problem (PDVRP) is a type of VRP where both deliveries and pickups need to be made between different locations. Various subtypes of PDVRP exist, such as VRP with Backhauls (VRPB), VRP with Clustered Backhauls (VRPCB), VRP with Mixed Backhauls (VRPMB), VRP with Divisible Deliveries and Pickups (VRPDDP), and VRP with Simultaneous Pickups and Deliveries (VRPSPD) (PARRAGH; DOERNER; HARTL, 2008; PARRAGH; DOERNER; HARTL, 2006). In the VRPSPD, for example, vehicles perform customer deliveries while also loading pickups simultaneously (MONTANé; GALVAO, 2006).

3. The VRP with Time Windows (VRPTW) is one of the most studied variants of VRP. In this type, each customer specifies a specific time interval during which their order must be delivered by a vehicle (BRANDãO, 2011). VRPTW can be categorized into two types: the VRP with Soft Time Windows (VRPSTW) allows deliveries to be made even after the time windows have passed, but with the imposition of a penalty for late deliveries (IQBAL; KAYKOBAD; RAHMAN, 2015), while the VRP with Hard Time Windows (VRPHTW) strictly prohibits any delays beyond the specified time windows (MIRANDA; CONCEIçãO, 2016).

4. The Split Delivery VRP (SDVRP) is a variant of the original VRP that relaxes the constraints regarding customer service. In SDVRP, customers' orders can be split and assigned to multiple vehicles on different routes, unlike the original VRP where each customer must be served exactly once and by only one vehicle (SILVA; SUBRAMANIAN; OCHI, 2015).

5. The Heterogeneous Fleet VRP (HFVRP) is a variant of the original VRP that introduces the concept of multiple types of vehicles with varying loading capacities (LEUNG et al., 2013).

6. The Periodic VRP (PVRP) is a variant of the VRP where customers have specific service requirements within defined time periods. In the PVRP, customers are first assigned to service patterns, which determine the days on which they require deliveries. Subsequently, a separate VRP is solved for each day within the planned time period, considering the delivery needs of all customers on that particular day (GULCZYNSKI; GOLDEN; WASIL, 2011).

7. The Green VRP (GVRP): Alongside the objective of minimizing transportation costs, the GVRP introduces the consideration of minimizing CO2 emissions for companies

(LIN et al., 2014). This problem arises from the need to address environmental concerns and promote sustainable transportation practices.

8. The Stochastic VRP (SVRP): In this variant, one of the parameters of the problem is represented as a stochastic variable that follows a known or unknown probability distribution. Different variations of SVRP include the VRP with stochastic demand (VRPSD), the VRP with stochastic customers (VRPSC), the VRP with stochastic demands and customers (VRPSDC), and the VRP with stochastic travel and service times (VRPSTS) (MARINAKI; MARINAKIS, 2016).

9. The Open VRP (OVRP): In contrast to the original VRP, the vehicles in the OVRP do not necessarily return to the central depot after completing their services (MARINAKIS; MARINAKI, 2014).

There exist additional variants of the VRP that are not as prevalent as the classes listed above. These include the Time-Dependent VRP (TDVRP), Multi-Trip VRP (MT-VRP), Dynamic VRP (DVRP), VRP with Loading Constraints (VRPLC), Truck and Trailer Routing Problem (TTRP), Multi-Compartment VRP (MCVRP), Fuzzy VRP, and Site-Dependent VRP (SD-VRP), among others (LIN et al., 2014).

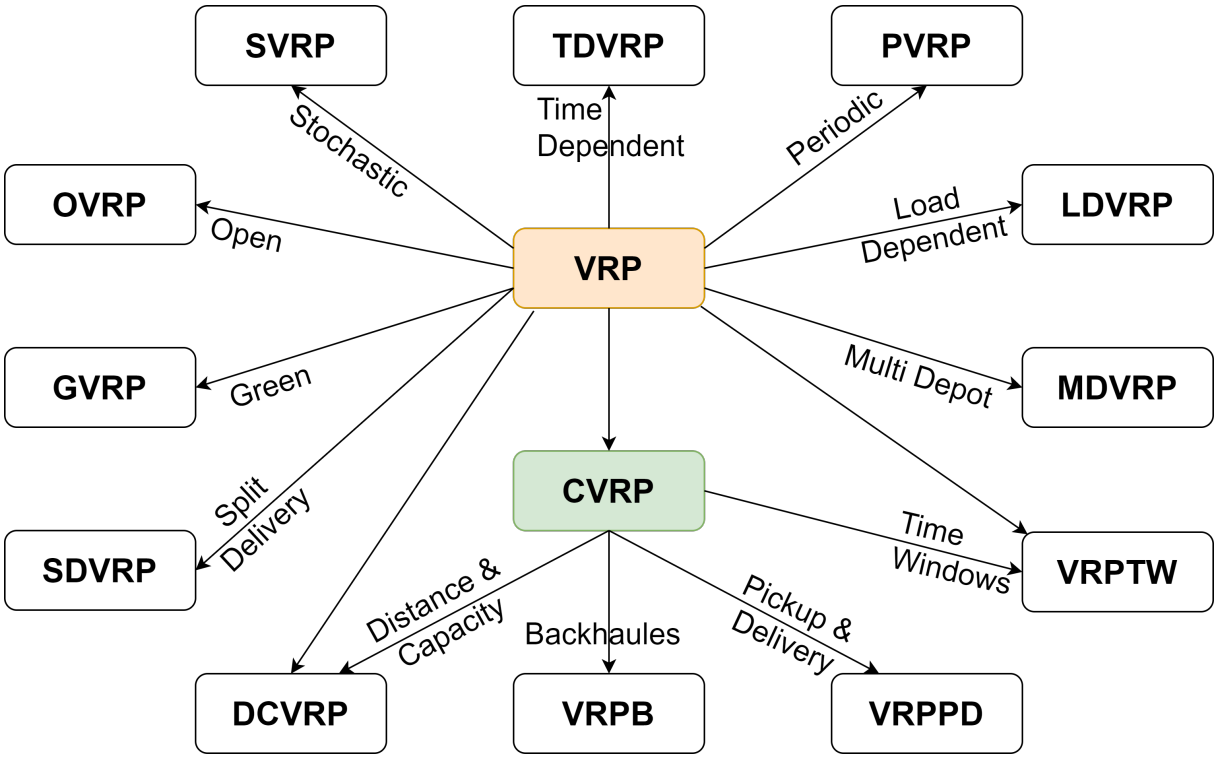

Figure 1 – Hierarchy of VRP variants

2.2 CVRP formulation

CVRP is defined as a complete undirected graph $G = (V, E)$. Set $V = \{0, 1, 2, \ldots, n\}$ is a set of nodes where the single central depot is represented by node 0 and $N = \{1, 2, \ldots, n\}$ is set of customers. The depot is a base for a fleet $K = \{1, 2, \ldots, |K|\}$. Set $E = \{e = \{i, j\} = \{j, i\} : i, j \in V, i \neq j\}$ is the edge set with nonnegative traveling cost c_{ij} between nodes i and j for each $\{i, j\} \in E$. The *fleet* is assumed to be homogeneous i.e. all $|K|$ vehicles are available at the *central depot*, have the identical capacity $Q > 0$ and all operate at equal cost. The objective is to determine the $|K|$ vehicle routes in such a way that (TOTH; VIGO, 2014):

- each route is started and finished at the central depot
- each customer $n \in N$ is visited exactly once
- the vehicle capacity Q is not exceeded by the total demand of customers in each single route
- the sum of the distances traveled by the vehicles to serve all the customers is minimized.

Assuming such constraints, the objective function may be defined as Equation (2.1).

$$objective\,function \equiv minimize(\sum_{k \in K} \sum_{(i,j \in E)} c_{ij} x_{ij}^{k}) \tag{2.1}$$

where for all $\{i, j\} \in E, x_{ij} = 1$ if a vehicle travels from i to j, otherwise $x_{ij} = 0$.

2.3 Solution algorithms for solving VRP

Given the complexity of the problem, there are two potential approaches for its resolution: exact methods and approximate methods. Exact methods aim to obtain optimal solutions and provide a guarantee of their optimality. However, for NP-complete problems, exact algorithms are nonpolynomial-time algorithms (unless $P = NP$). On the other hand, approximate methods, also known as heuristic methods, generate solutions of high quality within a reasonable time frame for practical applications. However, these methods do not offer a guarantee of finding a globally optimal solution (TALBI, 2009). Figure 2 illustrates the classification of algorithms that have been employed to address the Vehicle Routing Problem (VRP) and its variations.

2.3.1 Exact methods:

Within the class of exact methods, several classical algorithms can be identified. These include dynamic programming, a family of algorithms known as branch and X (branch and bound, branch and cut, branch and price), which were developed within the operations research community, constraint programming, and a family of search algorithms known as A* (A-star) and IDA* (iterative deepening algorithms), which were developed within the field of artificial intelligence. These enumerative methods can be regarded as tree search algorithms. The search process spans the entire search space of interest, and the problem is solved by dividing it into simpler subproblems (TALBI, 2009).

Exact methods are suitable for solving small instances of challenging problems. Due to the NP-hard nature of the Vehicle Routing Problem (VRP), exact methods are unable to consistently solve instances with more than 200 customers (TOTH; VIGO, 2014). It is important to note that the difficulty of a problem is not solely determined by the size of the instance, but also by its structure. For a given problem, there may be small instances that cannot be solved by an exact algorithm, while larger instances can be successfully solved using the same algorithm (TALBI, 2009). Among the most successful exact methods documented in the literature are the Branch and Bound algorithm (CHRISTOFIDES; EILON, 1969) and the Branch-and-cut algorithm (LYSGAARD; LETCHFORD; EGLESE, 2004).

2.3.2 Heuristics and Metaheuristics

Heuristics are employed to find "good" solutions for large-scale problem instances, offering acceptable performance at reasonable costs across a wide range of problems. Generally, heuristics do not provide an approximation guarantee for the solutions obtained. These methods can be classified into two categories: specific heuristics and metaheuristics. Specific heuristics are custom designed to tackle a particular problem or instance, while metaheuristics are versatile algorithms that can be applied to solve various optimization problems. Metaheuristics can be seen as high-level general methodologies that guide the design of specific heuristics for solving particular optimization problems. Unlike exact methods, metaheuristics enable the handling of large-scale problem instances by providing satisfactory solutions within a reasonable timeframe. However, there is no assurance of finding globally optimal or even bounded solutions (TALBI, 2009).

In recent years, researchers have proposed advanced mathematical programming decomposition algorithms to address the Vehicle Routing Problem (VRP). However, despite these efforts, only relatively small instances involving approximately 200 customers can be optimally solved, and there is significant variability in the computation times.

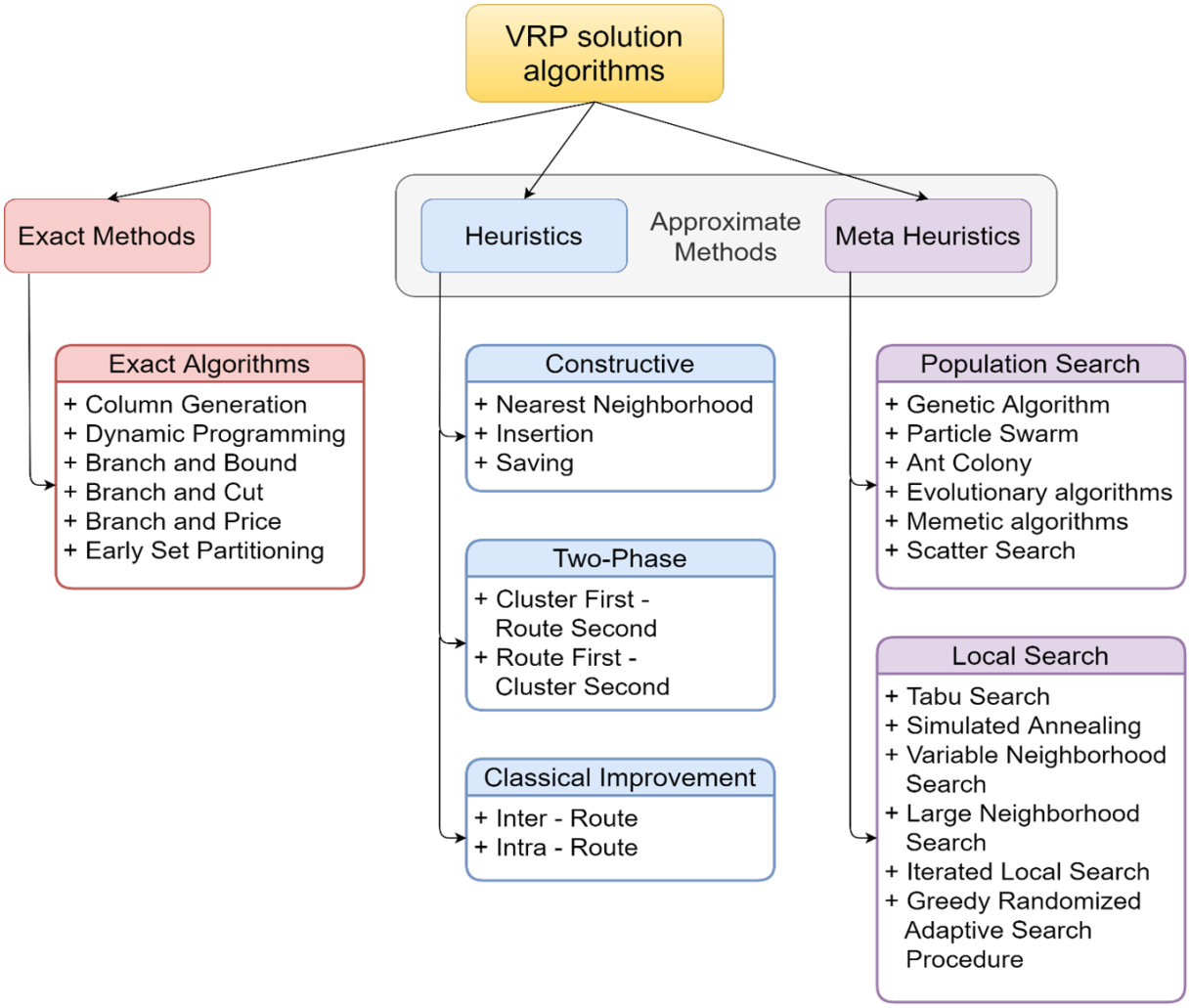

Figure 2 – Classification of VRP solution algorithms

In real-life scenarios, instances of the VRP are often large and require quick solutions within predictable times, making efficient heuristics essential in practice. Moreover, due to variations in the exact problem specifications across different settings, it becomes necessary to develop heuristics that offer sufficient flexibility to handle diverse objectives and side constraints. These concerns have been taken into account in the development of algorithms in the past few years, highlighting the need for heuristics that strike a balance between efficiency, flexibility, and problem complexity.

Heuristic algorithms can be broadly classified into three main categories: Constructive heuristics, Two-Phase heuristics, and Local Improvement heuristics. These categories encompass specific algorithms, as depicted in Figure 2.

Simple heuristics are primarily intended for constructing a single solution, whereas local search procedures and metaheuristics analyze multiple solutions, either by generating a sequence of solutions or by operating on a population of solutions. In addition to their application in metaheuristics for initializing solutions, simple heuristics are commonly employed in commercial VRP software to rapidly obtain feasible solutions of satisfactory

quality. The popularity of simple heuristics can be attributed to their simplicity and their adaptability to accommodate different types of constraints.

2.4 Heuristics

2.4.1 Constructive Heuristics

Nearest Neighborhood: Many constructive methods for the Capacitated Vehicle Routing Problem (CVRP) are derived by adapting heuristics from the Traveling Salesman Problem (TSP) to construct multiple routes. One of the simplest examples is the nearest neighbor heuristic. There are two main approaches for constructing the routes: sequential route building and parallel route building. In sequential route building, the process starts from the depot, and a route is gradually expanded by adding the nearest unrouted customer, provided that the customer's demand is compatible with the remaining capacity of the vehicle. Once no more customers can be added to the route, the vehicle returns to the depot, and a new route is initiated. With this approach, the routes are constructed one by one, sequentially. On the other hand, in parallel route building, the process begins from the depot, and each emerging route (1, 2, ..., $|K|$) serves the closest unrouted customer, resulting in $|K|$ routes each having one customer. Then, the $|K|$ routes are iteratively traversed again to accommodate their second customer, and so forth, until all customers are assigned to routes (TOTH; VIGO, 2014).

Both approaches provide different ways of constructing routes in the CVRP, and the choice between sequential and parallel route building depends on the specific requirements and constraints of the problem at hand.

Insertion heuristics: Insertion heuristics in the context of the Vehicle Routing Problem (VRP) involve initializing routes as empty loops starting from the depot, and then gradually inserting customers one by one into these routes. This insertion process can be performed either sequentially or in parallel, similar to the nearest neighbor algorithm. The primary objective of each insertion is to minimize the increase in route length, following the principle of cheapest insertion. The heuristic aims to identify the most cost-effective position within the existing routes to insert each customer, considering factors such as distance, travel time, or any other relevant cost metric. By minimizing the increase in route length during the customer insertion process, these heuristics contribute to constructing efficient and optimized solutions for the VRP (TOTH; VIGO, 2014).

Saving heuristics: In the Clarke and Wright heuristic, which is a type of saving algorithm (CLARKE; WRIGHT, 1964), the initial solution is formed with n routes, each containing only one customer, resulting in a complete but highly expensive solution. In each iteration of the algorithm, possible mergers or concatenations between two routes are

evaluated. The algorithm selects the merger that yields the largest positive saving or the smallest negative cost variation. For two routes to be merged, their combined total load must fit within the capacity of a single vehicle. The process of merging routes continues until a single route, representing a Traveling Salesman Problem (TSP) tour, is obtained, typically after $n - 1$ mergers. The merging process may also terminate if the merger of any two remaining routes would violate the capacity constraint of the vehicles.

2.4.2 Two-Phase heuristics

Cluster First – Route Second (CFRS): These methods employ a two-step approach to tackle the Vehicle Routing Problem (VRP). Initially, these methods create clusters or groups of customers, where the total demand of each cluster fits the capacity of a vehicle. Subsequently, a Traveling Salesman Problem (TSP) is solved for each cluster, treating each cluster as a subset of customers assigned to a single vehicle.

An exemplary method that demonstrates the CFRS approach is the sweep heuristic developed by Gillett and Miller (GILLETT; MILLER, 1974). In this heuristic, clusters are defined as angular sectors centered on the depot. The sweep starts at a certain angle and proceeds in a clockwise or counterclockwise direction, including customers within the defined angular sectors into the same cluster until the vehicle capacity is reached. Once the clusters are formed, a TSP is solved for each cluster individually.

Route First – Cluster Second (RFCS): These heuristics follow a different approach compared to Cluster First - Route Second methods. In RFCS heuristics, the first step involves relaxing the vehicle capacity constraint, allowing for a complete Touring Salesman Problem (TSP) solution where all customers are visited in a giant tour. The resulting giant tour includes all customers and does not consider vehicle capacity limitations. In the second step, a splitting procedure called Split is applied to divide the giant tour into multiple routes that satisfy the capacity constraints of the VRP.

In 2004, Prins introduced the idea of utilizing the Split procedure within a memetic algorithm for the Capacitated Vehicle Routing Problem (CVRP), where the candidate solutions are encoded as giant tours (PRINS, 2004). This innovation allowed for the development of the first evolutionary algorithm capable of competing with the state-of-the-art tabu search algorithms at that time.

2.4.3 Classical Improvements

Inter – Route: Inter-Route improvement moves play a crucial role in achieving high-quality results in practice. These moves involve modifications between different routes and are essential for optimizing solutions in the Vehicle Routing Problem (VRP) (TOTH;

VIGO, 2014). The inter-route improvement moves include several classical operators commonly used in the field. These operators consist of the following actions:

1. RELOCATE: This operator involves removing a consecutive sequence of k customers from their current route and reinserting them elsewhere in a different route. It allows for reorganizing the allocation of customers between routes.

2. SWAP: The SWAP operator swaps consecutive customers between different routes. This exchange helps to improve the overall arrangement of customers among the routes, potentially reducing total travel distances or balancing the workload across vehicles.

3. 2-OPT*: This operator focuses on modifying two edges from different routes. It entails removing two edges and reconnecting them differently, resulting in a new configuration. The objective is to optimize the connectivity and arrangement of customers across routes.

These three types of moves, along with the well-known 2-OPT operator, are among the most commonly used methods for inter-route improvement.

Intra – Route: Intra-Route improvement refers to the application of improvement heuristics designed for the Traveling Salesman Problem (TSP) within each individual route of the VRP. Any improvement heuristic that is originally designed for the TSP can be adapted and applied as an intra-route improvement method in the VRP. One example of an intra-route improvement heuristic is the $\lambda-Opt$ exchange, which was proposed by Lin (LIN, 1965). In the $\lambda-Opt$ exchange, a specified number of edges, denoted by λ, are removed from the route, and then replaced with λ other edges. This exchange helps to optimize the ordering of the customers within a single route, aiming to improve the overall route length and minimize travel distances.

2.5 Metaheuristics

Unlike exact methods, which are limited in their ability to handle large-scale problem instances, metaheuristics offer a viable approach to tackling such challenges by providing satisfactory solutions within a reasonable time frame. However, it is important to note that metaheuristics do not guarantee finding global optimal solutions or even bounded solutions. Nonetheless, metaheuristics have gained significant popularity in the past two decades due to their efficiency and effectiveness in solving large and complex problems (TOTH; VIGO, 2014).

Metaheuristics have found applications in various fields, including Engineering design, Machine learning and data mining, System modeling, Planning in routing problems, logistics, and transportation. Their versatility allows them to be adapted to different problem domains, making them valuable tools for addressing a wide range of real-world

problems. The success and widespread application of metaheuristics in various domains highlight their ability to provide practical solutions when exact methods are impractical or infeasible.

When designing a metaheuristic, it is crucial to consider two contradictory criteria: exploration of the search space (*diversification*) and exploitation of the best solutions discovered (*intensification*). These criteria are essential for effectively navigating the search process.

During the intensification phase, the metaheuristic focuses on exploring promising regions identified by the "good" solutions obtained thus far. This involves conducting a more thorough exploration within these regions in the hopes of finding even better solutions. The objective is to exploit the potential of these regions to improve the quality of the solutions.

On the other hand, diversification aims to ensure that the search process does not become confined to a limited number of regions. It involves visiting non-explored regions within the search space to achieve an even exploration of all regions. This prevents the search from becoming biased towards a specific subset of the search space, enhancing the chances of discovering new and potentially superior solutions.

By striking a balance between exploration and exploitation, metaheuristics can effectively search the search space, combining the strengths of both approaches (TALBI, 2009). Figure 3 provides a graphical representation of these concepts and illustrates how the metaheuristic navigates the search space, combining diversification and intensification strategies.

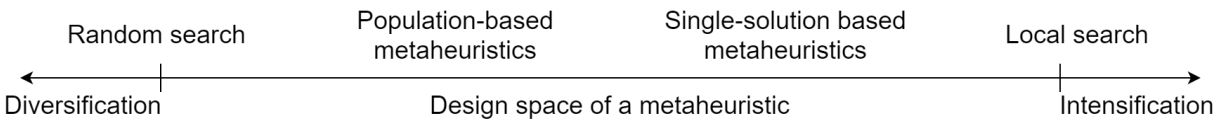

Figure 3 – Two conflicting criteria in designing a metaheuristic (TALBI, 2009)

The past 15 years have witnessed significant advancements in Vehicle Routing Problem (VRP) heuristics, primarily within the realm of metaheuristics. The evolution of VRP heuristics can be best described by the concept of hybridization. This evolution has occurred in three main aspects (TOTH; VIGO, 2014).

Firstly, new heuristics have emerged by combining different concepts that were initially developed independently. These concepts often relate to search principles such as simulated annealing, tabu search, variable neighborhood search, and genetic algorithms. By integrating these different approaches, researchers have been able to create powerful hybrid heuristics that leverage the strengths of multiple techniques.

Secondly, successful VRP methods have incorporated various other strategies, such as exotic large neighborhoods, exact mathematical techniques, decomposition approaches, and cooperation schemes. These additions have enhanced the performance and solution quality of the heuristics.

Thirdly, there has been a hybridization of scope, whereby researchers are now developing flexible methods (e.g., Cordeau et al. (CORDEAU; LAPORTE; MERCIER, 2001), Vidal et al. (VIDAL et al., 2014), and Subramanian et al. (SUBRAMANIAN et al., 2012)) that can be directly applied to solving a wide range of VRP variants without major structural changes. These flexible methods can address different problem variations efficiently, providing a versatile approach to VRP problem-solving.

Currently, metaheuristics for the VRP can be broadly classified into two main categories: Local Search methods, which focus on exploring and improving individual solutions, and population-based heuristics, which utilize a population of solutions to navigate the search space and find high-quality solutions.

2.5.1 Local Search algorithms

Local search methods explore the solution space by moving at each iteration from a solution to another solution in its neighborhood. As shown in Figure 2, this category includes Tabu Search (TS), Simulated Annealing (SA), Variable Neighborhood Search (VNS), Large Neighborhood Search (LNS), Iterated Local Search (ILS) and Greedy Randomized Adaptive Search Procedure (GRASP) that are described briefly as following:

Simulated Annealing (SA): The simulated annealing (SA) algorithm for combinatorial optimization was introduced by Kirkpatrick et al. in 1983 (KIRKPATRICK; JR; VECCHI, 1983) and is inspired by the annealing process in metallurgy. It uses a temperature parameter to control the acceptance of solutions that may worsen the objective function. SA allows for exploration of the solution space and can escape local optima. A well-known application of SA to the VRP is that of Osman (OSMAN, 1993).

Greedy Randomized Adaptive Search Procedure (GRASP): The GRASP was initiated by Feo and Resende in 1989 (FEO; RESENDE, 1989). The GRASP principle is to create a new solution at each iteration, independent of the following. For this purpose, two stages are necessary: first, a solution is built using a randomized greedy algorithm and, second, the obtained solution undergoes a local search. The best solution found through the iterations is returned as the result.

Tabu Search (TS): Tabu Search (TS) was developed by Glover (GLOVER, 1986; GLOVER, 1989; GLOVER, 1990) and has been widely applied to various combinatorial problems. Unlike GRASP and SA, TS is a fully deterministic method in the context of VRP. It has proven to be highly effective in solving difficult combinatorial problems and

often produces very good solutions.

The fundamental principle of TS is to continue the local search even when a local optimum is reached. This means that TS aims to thoroughly explore the neighborhood of the current solution and make the best possible move, even if it results in a temporary deterioration of the objective function. The key idea behind TS is to prevent revisiting previously explored solutions using memories called tabu lists.

Tabu lists are used to record the recent history of the search and ensure that the search process does not return to previously visited solutions. This memory mechanism guides the search and promotes diversification by preventing cycling and forcing exploration of new regions within the solution space.

In the context of the Vehicle Routing Problem (VRP), Taillard (TAILLARD, 1993) proposed the Tabu Search algorithm as a method specifically tailored to solve VRP instances. TS has been successfully applied to VRP and has demonstrated its effectiveness in finding high-quality solutions for vehicle routing problems.

Variable Neighborhood Search (VNS): The Variable Neighborhood Search (VNS) and its simpler variant, the Variable Neighborhood Descent (VND), are highly efficient and concise metaheuristics. These approaches are often utilized as replacements for local search procedures within other metaheuristics. The concept of Variable Neighborhood Search was introduced by Mladenović and Hansen (MLADENOVIć; HANSEN, 1997).

The main idea behind Variable Neighborhood Search is inspired by a straightforward principle: systematically changing the neighborhood each time a local search algorithm fails to find an improvement in the current neighborhood. This process of systematically exploring different neighborhoods allows for a more extensive search of the solution space, potentially leading to better solutions.

In the context of VRP, Kytöjoki et al. (KYTöJOKI et al., 2007) successfully applied VNS to tackle this problem. Their proposed algorithm incorporated seven frequently used operators: 2-opt, Or-opt, and 3-opt as intra-route moves, and exchange, relocate, 2-opt*, and cross-exchange as inter-route moves.

Iterated Local Search (ILS): Iterated Local Search (ILS) appeared in the literature a few years before Variable Neighborhood Search (VNS) and was initially introduced by Baum in 1986 under the name "iterated descent" (BAUM,). The ILS approach involves searching the solution space by exploring the attraction basin around a solution using local search before moving away from it using perturbation. ILS consists of four main components: generation of an initial solution, local search, perturbation and, acceptance criterion. Vansteenwegen et al. (VANSTEENWEGEN et al., 2009) proposed an ILS approach for a specific vehicle routing problem called the Team Orienteering Problem with Time Windows. In this problem, each customer is associated with a score, service

time, and time window. The objective is to maximize the total score collected by a fixed number of routes while respecting the time windows.

Large Neighborhood Search (LNS): The Large Neighborhood Search (LNS) framework was introduced by Shaw in 1998 (SHAW,). It is a search strategy that focuses on exploring a large neighborhood of solutions, which increases the likelihood of finding improved solutions compared to locally optimal solutions. LNS efficiently searches this large neighborhood, enabling effective exploration of the solution space. The key idea behind LNS is to decompose the original problem by unfixing certain decision variables, resulting in a partial solution. By unfixing these variables, a neighborhood of solutions is defined, which can be explored efficiently using a specific procedure such as a heuristic or a mixed-integer programming (MIP) solver. This procedure rapidly searches the defined neighborhood, aiming to find an improved solution.

If the procedure discovers an improved solution within the neighborhood, it replaces the current solution, and a new large neighborhood is defined around this new solution. This process of defining and searching large neighborhoods is repeated iteratively until a stopping criterion is met, such as a maximum number of iterations or reaching a desired solution quality.

2.5.2 Population-based algorithms

Population-based heuristics involve the evolution of a population of solutions, where combinations and interactions among individuals aim to generate improved solutions. This category encompasses various metaheuristics, including Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Evolutionary Algorithms (EAs), Memetic Algorithms (MAs), Scatter Search (SS), and Path Relinking (PR). Figure 2 illustrates the classification of these heuristics.

Genetic Algorithms (GAs): Genetic Algorithms (GAs) are population-based metaheuristics that emulate biological evolution processes to solve problems and model evolutionary systems. The concept of GAs was first introduced by Holland in 1973 (HOLLAND, 1973). The evolution process in GA comprises three major operators: selection, crossover, and mutation.

In the selection step, two parents are chosen from the population, with a preference given to the best parents based on their fitness. The parents are then subjected to the crossover operator, which combines their genetic material to generate offspring solutions. The purpose of crossover is to exchange genetic information and create diverse offspring. Additionally, the mutation operator introduces random modifications to the child solution with a certain probability, ensuring population diversity.

While pure GA implementations have demonstrated mixed results in various

problem domains, notable success has been achieved in the vehicle routing problem with time windows (VRPTW) in the works of (THANGIAH, 2019) and (POTVIN; BENGIO, 1996). These studies have specifically tailored the GA approach to address the challenges posed by VRPTW.

A comprehensive description of the GA can be found in Section 2.7 of this chapter, where the workings and intricacies of the algorithm are discussed in detail.

Memetic Algorithms (MAs): Memetic Algorithms (MAs), initially proposed by Moscato (MOSCATO, 1989), represent more advanced versions compared to Genetic Algorithms (GA) by incorporating a local search procedure for improving the performance on various optimization problems. The inclusion of local search in MAs contributes to intensification (MOSCATO, 1999), as the solutions resulting from crossover undergo a local search before undergoing mutation.

An enhanced variant of MAs called Memetic Algorithm with Population Management (MA—PM) was introduced by Sörensen and Sevaux in 2006 (SöRENSEN; SEVAUX, 2006). MA—PM incorporates a strategy to maintain population diversity by accepting new solutions only if they exhibit sufficient dissimilarity from the existing solutions in the population.

The first successful application of MAs to the Vehicle Routing Problem (VRP) was proposed by Prins (PRINS, 2004). The method combines genetic operators such as selection and crossover with an efficient local search procedure, which replaces the conventional randomized mutation operator. This integration of different components makes the MA approach highly effective for solving the VRP.

Scatter Search (SS): Scatter Search is a population-based metaheuristic that incorporates diversification strategies similar to tabu search. In contrast to Genetic Algorithms (GA), SS typically operates on a smaller set that contains a collection of good-quality and diverse solutions. This approach, initially introduced by Glover in 1977 (GLOVER, 1977), maintains a pool of high-quality solutions encountered during the search process. It also ensures diversity by including well-scattered solutions to facilitate exploration of alternative regions when the optimization process becomes trapped in a local minimum.

One notable successful application of SS is proposed by Russel and Chiang (RUSSELL; CHIANG, 2006) for addressing the vehicle routing problem with time windows.

Path Relinking (PR): Path Relinking is a metaheuristic approach that differs from traditional genetic algorithms as it does not utilize crossover operators. Instead, it focuses on utilizing two solutions from a population to generate new solutions. Path Relinking is an evolutionary search strategy that explores the trajectory connecting two high-quality solutions. The concept of Path Relinking was initially proposed by Glover as a means to intensify tabu search or Scatter Search (GLOVER, 1997; GLOVER; LAGUNA;

MARTI, 2000). Path Relinking can be considered an evolutionary method since it generates new solutions by combining elements from existing solutions. However, unlike the genetic algorithm's family, randomness does not play a crucial role in generating offspring solutions. Path Relinking generates new solutions by systematically exploring paths that connect elite solutions, emphasizing the exploitation of promising regions within the solution space. Path Relinking serves as a powerful tool for intensification, utilizing the trajectory between high-quality solutions to guide the search process.

An application of Path Relinking in the context of the Capacitated Vehicle Routing Problem (CVRP) was carried out by Ho and Gendreau (HO; GENDREAU, 2006). They incorporated Path Relinking within a tabu search method specifically designed for the CVRP.

Ant Colony Optimization (ACO): The Ant Colony Optimization (ACO) approach (DORIGO; STUTZLE, 2004) draws inspiration from the social behavior of ants and their pheromone-based strategies during food foraging. In the ACO algorithm, ants initially deposit pheromone randomly to communicate and inform each other about food sources. Over time, the amount of pheromone deposited on specific paths increases if more ants follow those paths, while it diminishes if fewer ants pass through them. This pheromone-based communication system enables efficient exploration and exploitation of the solution space. ACO has emerged as one of the most widely used swarm-based algorithms for solving combinatorial optimization problems. It is particularly well-suited for problems where constructing a solution can be likened to creating paths within a given graph, as is the case in vehicle routing problems.

In the context of the Capacitated Vehicle Routing Problem (CVRP), Bullnheimer et al. (BULLNHEIMER; HARTL; STRAUSS, 1999) applied ACO to find optimal or near-optimal solutions. By leveraging the graph-like nature of the problem and mimicking the pheromone-based communication among ants, ACO offers a promising approach for solving the CVRP.

Particle Swarm Optimization (PSO): Particle Swarm Optimization (PSO) is a population-based metaheuristic that revolves around a group of individuals referred to as particles. It was initially introduced by Kennedy and Eberhart in 1995 (KENNEDY,) for continuous optimization problems and has since been adapted for discrete optimization problems (KENNEDY; EBERHART,). Each particle in PSO represents a potential solution and moves within the search space with the objective of reaching the global optimum. The learning process in PSO involves local interactions between the particles within the swarm. Each particle maintains a memory of the best solution it has encountered so far and can communicate with neighboring particles. Using this information, a velocity is computed for each particle, which determines the adjustments made to its position in the next iteration.

The application of PSO to vehicle routing problems was first presented by Chen et al. (CHEN; YANG; WU, 2006). In their work, they proposed a hybrid approach that combines PSO with simulated annealing. Specifically, the TSP (Traveling Salesman Problem) is solved using simulated annealing for each vehicle, while the assignment of customers to vehicles is conducted within the PSO framework.

2.6 Imperialist Competition Algorithm (ICA)

ICA was introduced as a new evolutionary algorithm by Atashpaz-Gargari in 2007. In contrast to other evolutionary algorithms, the ICA is based on the sociopolitical process of imperialism (ATASHPAZ-GARGARI; LUCAS,). As the original ICA, the algorithm is defined as following steps:

Step 1: Create initial empires:

The initial population of size N_{pop} provides an initial collection of countries. N_{imp} of the most powerful countries (based on fitness value) are selected to form the initial empires. The remaining N_{col} of the countries ($N_{col} = N_{pop} - N_{imp}$) are regarding as colonies that are divided amongst the N_{imp} empires such that the size of each empire – the number of colonies assigned; is relative to its imperialist power (relative fitness) – see Figure 4. As the color coding highlights, a larger star (higher imperialist power / fitness) has more colonies.

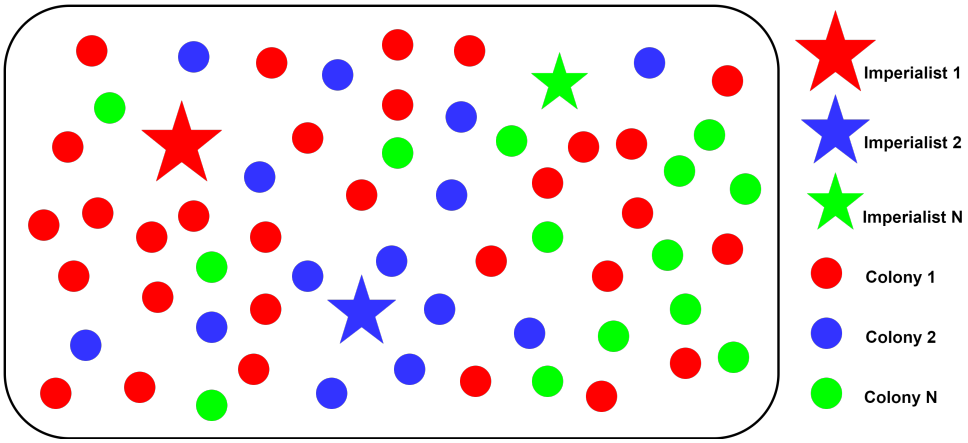

Figure 4 – Initializing the empires (inspired by (ATASHPAZ-GARGARI; LUCAS,))

Step 2: Assimilation: Moving colonies toward the imperialists:

The imperialist in each empire has the best fitness value in its corresponding empire. Therefore, each one is acting as a "Local Optima" in their empires. In this step, the imperialists start improving their corresponding colonies through crossover. The aim of this step is to search for points in the solution space near the imperialist (local optima),

providing improved fitness. Figure 5 illustrates the update process: where d is the distance between the colony and its imperialist. The new position of the colony depends on x (number of units to move) and θ (deviation from the direction to the empire). x is a random variable with uniform distribution where $x \sim U(0, \beta \times d)$ and β is greater than 1. θ, is a random number with uniform distribution (or any suitable distribution that provides a random amount of deviation). Therefore, $\theta \sim U(-\gamma, \gamma)$ where γ is a parameter that adjusts the deviation from the original direction.

Step 3: Revolution: Revolve the colonies:

Revolution is equivalent mutation in a GA. The "Revolution Rate" is thus a parameter that determines the percentage of the colonies that need to be revolved. First, the number of the revolved colonies are calculated by Equation (2.2), then the Revolution operator is applied to all the revolving colonies which are selected randomly.

$$NumOfRevolvedColonies = RevolutionRate \times NumberOfColoniesOfEmpire \quad (2.2)$$

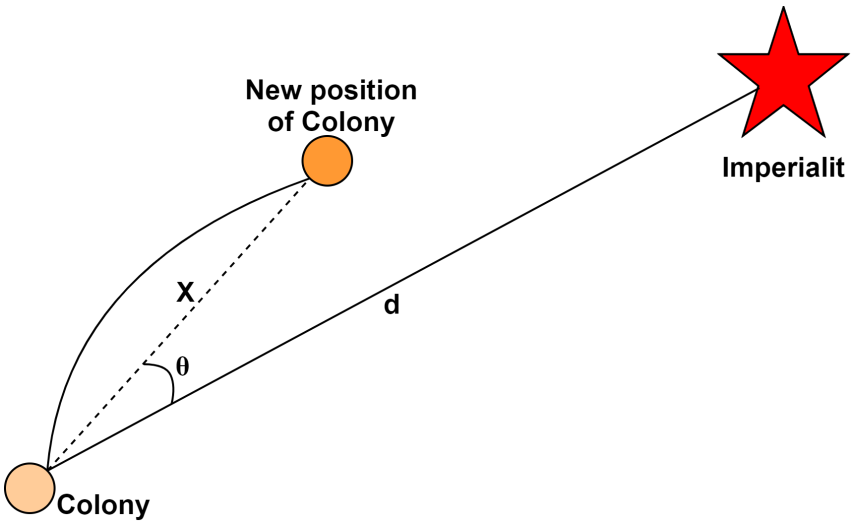

Figure 5 – Colony movement towards its imperialist (inspired by (ATASHPAZ-GARGARI; LUCAS,))

Step 4: Possess Empires: Swapping the position of colony and imperialist:

Moving the colonies towards their corresponding imperialist causes the colonies to reach new positions. Therefore, their new fitness value will be calculated. If one of the colonies has better fitness value than its corresponding imperialist, their positions will be exchanged as highlighted in Figure 6-(a) (before) and Figure 6-(b) (after) i.e. the colony with the best fitness will become the new Imperialist.

Step 5: Total cost calculation:

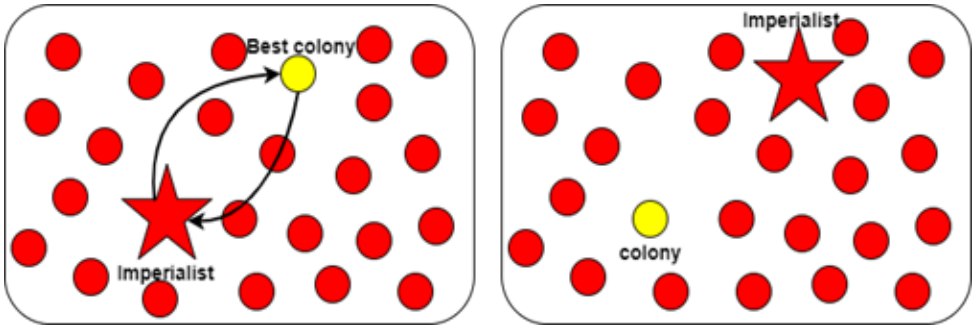

Figure 6 – a) Best colony and imperialist. b) After swap (inspired by (ATASHPAZ-GARGARI; LUCAS,))

The Total cost of each empire indicates the power of an empire, as shown in Equation (2.3):

$$T.C._n = Cost(imperialist_n) + \xi mean\{Cost(colonies\, of\, empire_n)\} \tag{2.3}$$

where $T.C._n$ is the total cost of the n^{th} empire, and $0 < \xi < 1$ is a parameter that defines the contribution factor of colonies to the power of an empire. The Cost represents the fitness function (or objective function).

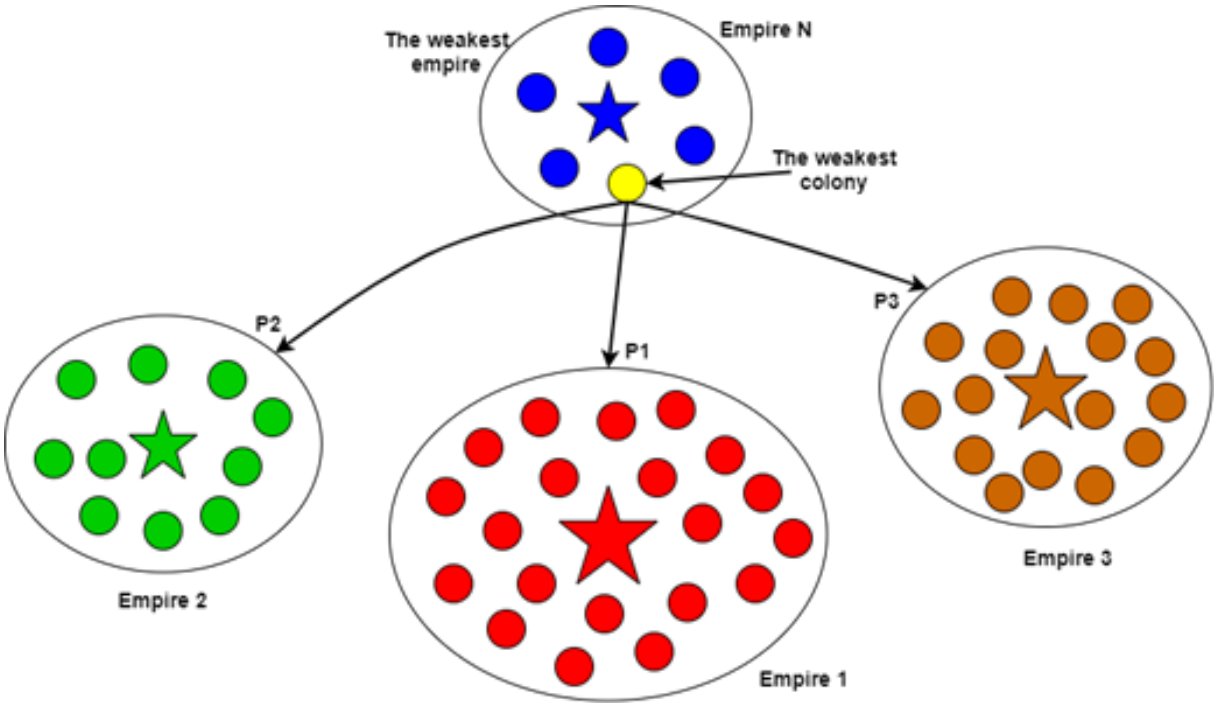

Figure 7 – An overview of the imperialist competition (inspired by (ATASHPAZ-GARGARI; LUCAS,))

Step 6: Imperialist competition:

In this competition between empires, the weakest colony of the weakest empire is selected for possession. An empire with more power has more likelihood to seize the colony.

If an empire has no colonies left, the imperialist itself is selected as a colony for possession and the empire ceases to exist. Figure 7 illustrates the imperialist competition where the size of stars again reflects the power of the imperialists and where the yellow colony represents the selected weakest colony. $P1$, $P2$ and $P3$ are the possession probabilities for empires 1, 2 and 3 calculated based on total cost of each empire as follows:

$$N.T.C._n = T.C._n - \mathbf{max}_i\{T.C._i\} \tag{2.4}$$

$$P_{P_n} = \left| \frac{N.T.C._n}{\sum_{i=1}^{N_{imp}} N.T.C._i} \right| \tag{2.5}$$

where $N.T.C._n$ is the normalized total cost of the n^{th} empire and $T.C._n$ is the total cost of the n^{th} empire (see Equation (2.3)). If $P = \{P_1, P_2, P_3\}$ and $R = \{r_1, r_2, r_3\}$ (uniformly distributed random numbers) then $D = P - R = \{P_1 - r_1, P_2 - r_2, P_3 - r_3\}$. The empire whose relevant index in D is maximum, will be the winner of this competition.

Stopping conditions:

The above-mentioned steps (2 to 6) are repeated iteratively. The weaker empires will collapse gradually by losing their colonies. Finally, only the most powerful empire will remain and its imperialist that is the solution's "Local Optima" i.e. it will be the best individual.

2.7 Genetic Algorithm (GA)

The Genetic Algorithm (GA) is an evolutionary optimization algorithm rooted in the principles of natural selection and biological reproduction. It draws inspiration from the Darwinian theory of evolution and the concept of survival of the fittest. The development of GA can be attributed to John Holland, a Professor of Electrical Engineering and Computer Science at the University of Michigan, and his colleagues in 1960. Their groundbreaking publication, "Adaptation in Natural and Artificial Systems," was released in 1975 by MIT Press (HOLLAND, 1973). Subsequently, David E. Goldberg further extended and refined the algorithm, publishing his book on genetic algorithms in 1989 (GOLDBERG D. E., 1989).

Genetic algorithms replicate the evolutionary mechanisms that have occurred in nature over millions of years. By incorporating principles from natural genetics, GA offers a means to solve complex problems using simple techniques within a finite timeframe. It leverages heuristics and historical information related to the problem domain to evolve towards improved solutions. Mimicking the evolutionary behavior of humans, GA involves the exchange of genetic material encoded as strings, tailored to the problem at hand (WHITLEY, 1994).

The simplicity and low computational complexity of GA have led to its widespread application in various domains, including engineering design, business, finance, and other complex scientific fields. GA's ability to handle complex problems with straightforward methods has made it an attractive choice in numerous practical applications.

2.7.1 Basics of Genetic Algorithm

The breakthrough in the development of the Genetic Algorithm (GA) occurred in the 1960s when John Holland successfully devised codes to represent genetic information. Initially, this approach employed binary strings, where each bit could denote a specific characteristic or feature. A value of 1 indicated the presence of the characteristic, while 0 indicated its absence. The system was initially designed as a classifier, where the accuracy of classification depended on the fitness value of the encoded string. Strings with higher fitness values, corresponding to better classification results, would survive and be retained, while strings with lower fitness values would perish. With each generation, the strings undergo evolution, resulting in a population of strings with higher average fitness values, indicating improved quality (MITCHELL, 1995).

At the core of the GA is the chromosome, which constitutes the basic genetic material. The chromosome is composed of genes, and the alleles represent the different values that genes can take on. Each gene is a fundamental element of the chromosome. Figure 8 provides a visual representation of the essential components of the GA.

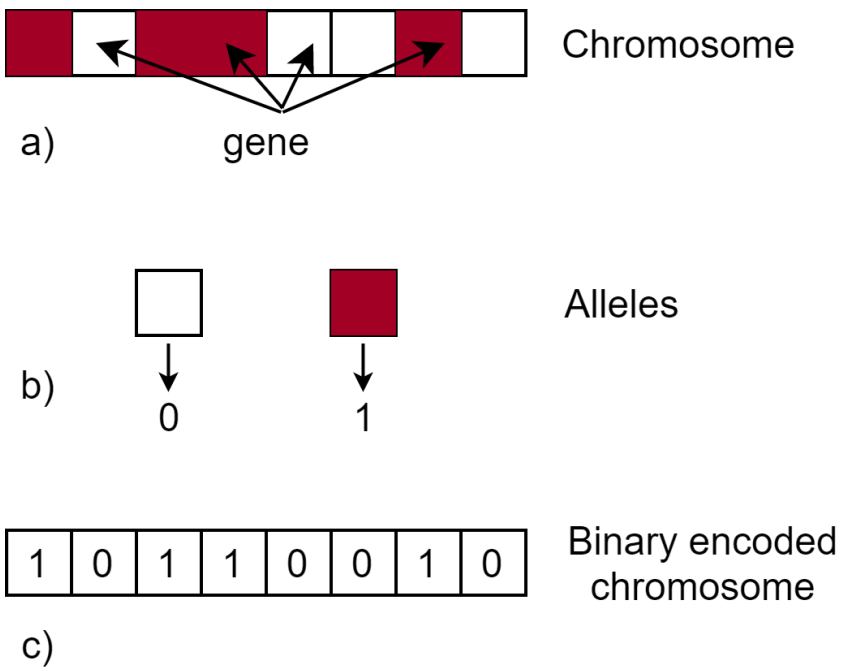

Figure 8 – a) Typical chromosome structure, b) Alleles, c) Binary encoded chromosome

2.7.2 Genetic operators

The Genetic Algorithm (GA) encompasses three crucial operators that play a pivotal role in its algorithmic design: selection, crossover, and mutation (REEVES, 2010). Each of these operators is described in detail below:

Selection: is a fundamental operation within the Genetic Algorithm (GA) that involves choosing individuals from the population for mating and reproduction through recombination. The selection process incorporates a degree of randomness, as there is no deterministic rule governing the selection of chromosomes for crossover or mutation. Three commonly employed selection methods include roulette wheel selection, tournament selection, and elitism (ABRAHAM; NEDJAH; MOURELLE, 2006).

Roulette wheel selection utilizes a rotating wheel divided into multiple regions, with each region corresponding to an individual in the population (BODENHOFER, 2003). The size of each region is proportional to the probability of selecting the corresponding individual, which, in turn, depends on their fitness. This selection method allows for a probabilistic approach to determine the parents for the next generation.

Tournament selection involves randomly selecting a subset of individuals from the population (referred to as the tournament size). From this chosen subset, the individuals with the highest fitness are selected as parents. This method creates a competitive environment where individuals with higher fitness have a greater chance of being chosen as parents.

Elitism is a selection strategy where the chromosome with the highest fitness value is directly copied into the next generation. By preserving the fittest individual, elitism ensures that their genetic material remains unaltered during the crossover or mutation process.

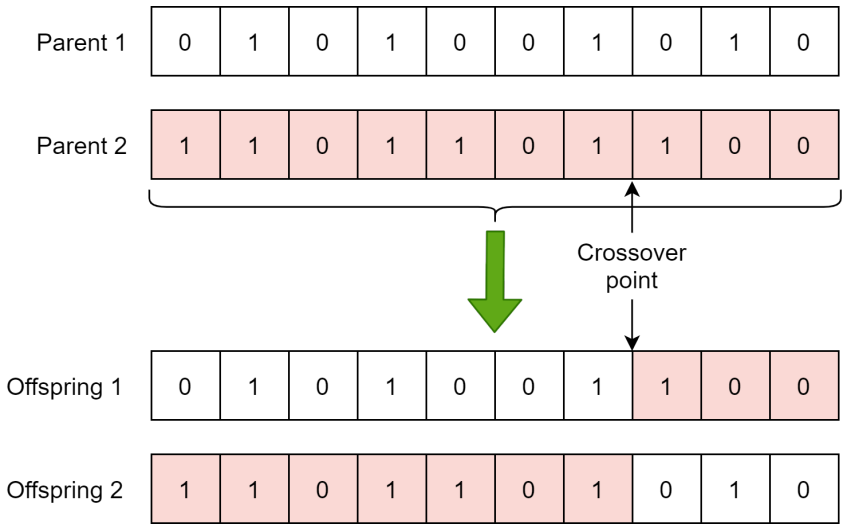

Figure 9 – One-point crossover

Crossover: is a vital operator within the Genetic Algorithm (GA) that generates offspring from two selected parents for recombination. Crossover can take the form of one-point, two-point, or multi-point operations.

In one-point crossover, a single bit in the two parent chromosomes is randomly chosen, and the segments on either side of the crossover point are exchanged, resulting in two offspring (Figure 9). This process facilitates the exchange of genetic material between the parents and introduces diversity into the population.

Two-point crossover involves selecting two points within the parent chromosomes randomly. The segment between these two points is then exchanged between the parents, leading to the creation of two offspring (Figure 10). This type of crossover promotes the mixing of genetic information from different regions of the parent chromosomes, aiding in the exploration of the solution space.

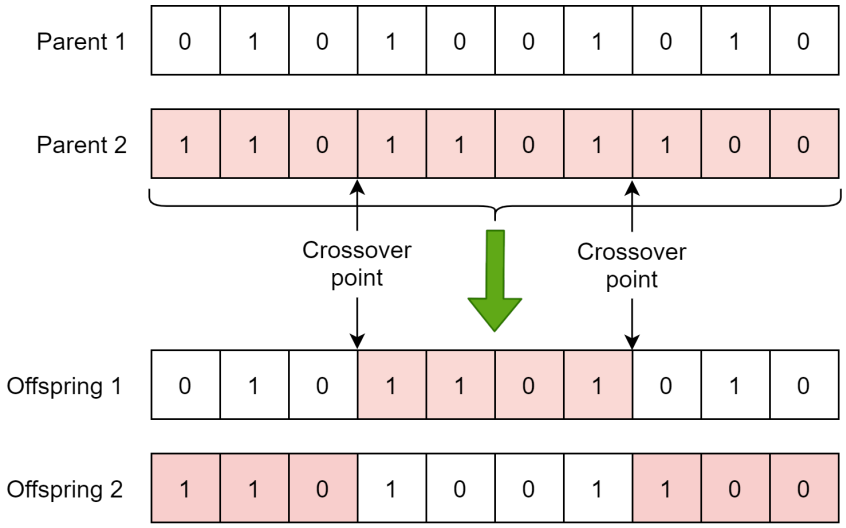

Figure 10 – two-point crossover

The choice between one-point, two-point, or multi-point crossover depends on the specific problem and the desired balance between exploration and exploitation of the solution space. Each crossover method offers its own advantages in terms of the diversity and quality of the offspring generated.

Mutation: is a crucial operator within the Genetic Algorithm (GA) where one or more bits in the chromosome string are subject to flipping with a fixed probability known as the mutation rate. This operation introduces diversity within the population and effectively expands the search space. Figure 11 illustrates the process of mutation in a chromosome encoded using binary representation.

During mutation, specific bits in the chromosome string are randomly selected and inverted, leading to alterations in the genetic information. By introducing these random changes, mutation serves as a mechanism to explore new regions of the solution space that

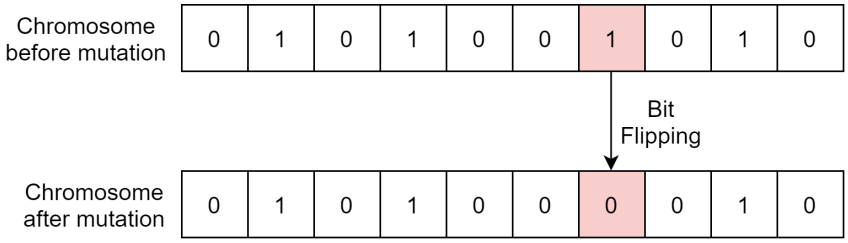

Figure 11 – Mutation operator

may not be easily accessible through selection and crossover alone.

The mutation rate, typically defined as a small value, governs the probability of a bit undergoing mutation. Higher mutation rates result in more extensive exploration, while lower mutation rates prioritize exploitation of the existing genetic material.

2.7.3 Algorithm

The Genetic Algorithm (GA) is a population-based metaheuristic that combines principles from natural selection and genetics. It aims to search for the optimal solution within the search space by incorporating randomness in the exploration process. GA operates as an iterative algorithm that converges towards an optimum solution over a finite number of iterations, with candidate solutions evolving towards higher fitness or quality in each iteration or generation. The termination of the algorithm occurs when either the optimum solution is achieved (convergence), the maximum number of iterations (generations) is reached, or a predefined stopping condition is satisfied.

The pseudocode for the common form of GA is presented in Algorithm 1. It outlines the key steps involved in the GA algorithm, including the initialization of the population, the evaluation of fitness, the iterative process of selection, crossover, and mutation, and the subsequent update of the population based on the fitness of the offspring generated.

2.8 Island Genetic Algorithm

Distributed Evolutionary Algorithms (EAs) are designed to distribute computing tasks across multiple processors or computing nodes. Two common approaches used in distributed EAs are the population-distributed model and the dimension-distributed model. In the population-distributed model, individuals or subpopulations from the evolutionary population are distributed among multiple processors or computing nodes. This allows for parallel processing of the population. Several variants of the population-distributed model exist, including the master-slave model, island model (or coarse-grained model), cellular model (or fine-grained model), hierarchical model (or hybrid model), and pool

Algorithm 1 Genetic Algorithm pseudo code

```
Select initial population of size N
Calculate the fitness values of the entire population
while ite ≤ MaxIter do
    Selection: select parent for reproduction
    Crossover: apply crossover on parents to produce offsprings
    Mutation: apply mutation on selected chromosoms (optional)
    Calculate fitness values of the population
    Select members of the next generation based on fitness values
    if termination condition met then
        exit
    else
        continue
    end if
end while
return chromosome with the best fitness
```

model. The master-slave model involves a central controller (master) that distributes tasks to multiple processors (slaves) for evaluation and processing. The island model divides the population into subpopulations that evolve independently on different processors, occasionally exchanging individuals to maintain diversity. The cellular model further divides the population into smaller groups (cells) that interact with neighboring cells through local exchanges. The hierarchical model combines different levels of subpopulations, where each level can have its own evolutionary dynamics. The pool model utilizes a shared pool of individuals that processors can access and modify during the evolutionary process. Figure 12 depicts the classification of Distributed Evolutionary Algorithms.

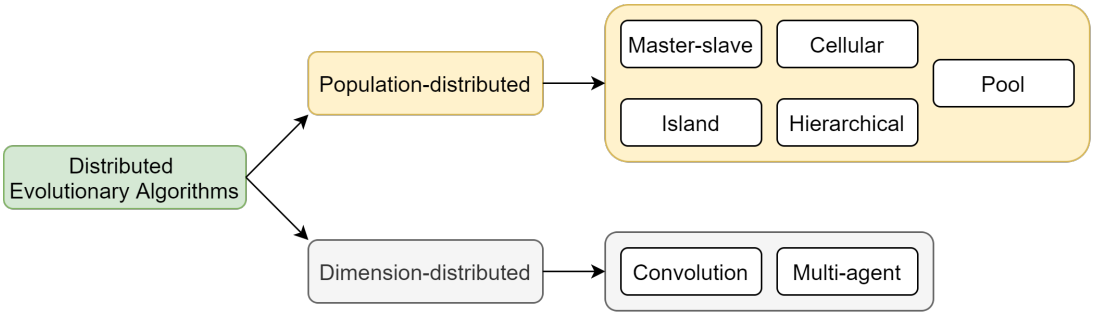

Figure 12 – Classification of Distributed EAs

The primary distinction between an island model and a single population model lies in the division of individuals into islands. Islands interact through migration, which serves as the sole means of communication between them. Migration plays a crucial role in the island model, as without it, the islands would function as separate, independent runs. Thus, the inclusion of migration is vital (CANTú-PAZ, 2001). The standard approach to implementing migration involves sending a specific number of individuals (migration size) at regular intervals (migration interval). These two parameters, migration size and interval,

are considered the most significant migration parameters as they control the quantitative aspect of migration, specifically the total number of individuals transmitted during the execution of the evolutionary algorithm. While alternative models exist where migrations are triggered based on certain criteria (BRANKE J.,), fixed schedules are more commonly employed and easier to analyze. Figure 13 provides an illustrative example of the scheme employed in the Island model, demonstrating the flow of migration between the islands.

Another crucial parameter in migration is the migration policy, which determines the selection and replacement strategy for the individuals being sent between populations. A common policy involves selecting the best individuals from the source population and replacing the worst individuals in the target population. However, Cantú-Paz has demonstrated that this approach results in an elevated selection pressure within the system (CANTú-PAZ, 2001). Alternatively, the random-random policy can be employed, where a random individual is chosen from the source population and replaces another random individual in the target population. This policy helps avoid an increase in selection intensity, making it suitable for investigating other migration parameters. It provides a more diverse exploration of the solution space without imposing additional selection pressure. It is worth noting that various other migration policies are also possible, depending on the specific requirements and characteristics of the problem being addressed. The choice of the migration policy should be carefully considered to strike a balance between maintaining diversity and promoting effective information exchange between populations.

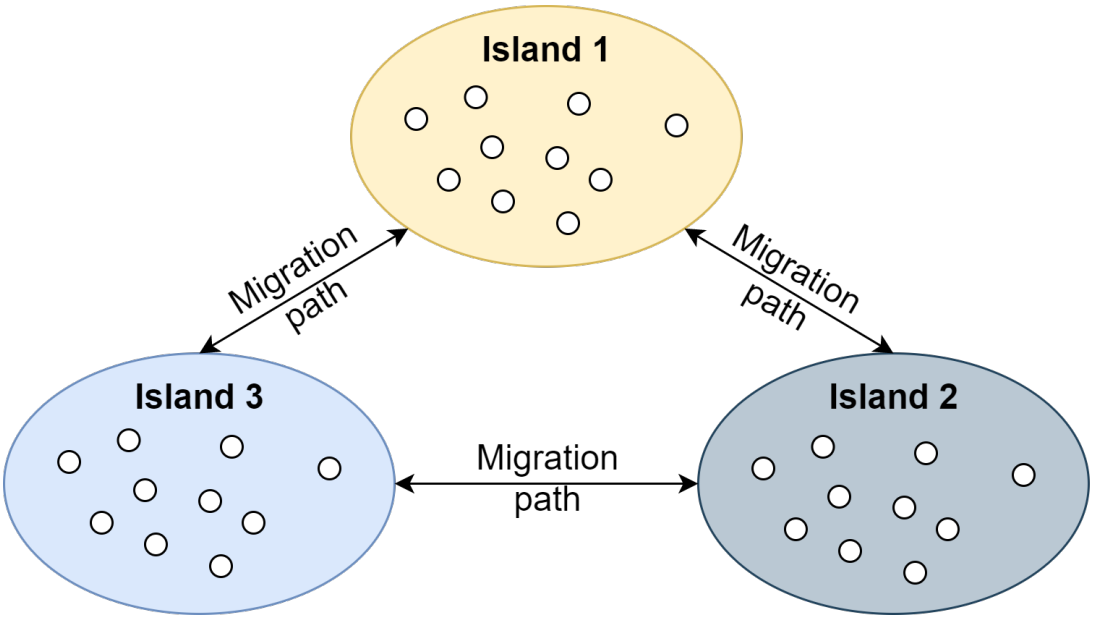

Figure 13 – an example of Island model GA scheme

The migration topology in the context of island models determines the pattern of migration, specifying which islands send individuals to which other islands. Several commonly used migration topologies include the array, ring, grid, star, and fully connected topologies. In the array topology, islands are arranged in a linear structure, with each island

having designated neighbors to which individuals can migrate. The ring topology forms a circular structure, where islands are connected in a loop. In the grid topology, islands are arranged in a two-dimensional grid, allowing for migration among neighboring islands in horizontal and vertical directions. The star topology consists of a central island connected to multiple satellite islands. Lastly, the fully connected topology establishes direct migration links between all pairs of islands. The choice of migration topology influences the spread and exchange of information among islands. Topologies with sparser connections, such as the array or ring, result in slower dissemination of information across the system. In contrast, topologies with denser connections, like the fully connected topology, facilitate faster and more extensive information flow among islands.

2.9 Hybrid Genetic Search for CVRP (HGS-CVRP)

HGS-CVRP is a specialized version of the UHGA (Unified Hybrid Genetic Search) for solving CVRP, through the introduction of the new neighborhood search, SWAP* (VIDAL, 2022). Its search schema is the same as its original method (UHGA - (VIDAL et al., 2012)) which is a combination of 3 strategies:

1. **Combining crossover and a neighborhood-based search:** Crossover improves diversification through more exploration in the solution space and the neighborhood-based search provides aggressive solution improvement.
2. **A controlled exploration of infeasible solutions:** which provides more exploration in the areas close to the feasibility boundaries.
3. **Advanced population diversity management strategies:** which allows the population to be diversified with high quality solutions.

As depicted in Figure 14, HGS-CVRP performs following the steps until a stopping condition is met:

Step 1 – Parents Selection: To select each parent, binary tournament selection is applied, returning an individual (solution) with the best fitness value of the tournament. To calculate the fitness value, two parameters are considered: objective value and diversity. Therefore, each individual has two ranking characteristics: in terms of solution quality and in terms of diversity contribution. The fitness value is calculated based on a weighted sum of these ranks.

Step 2 – Recombination: An ordered crossover (OX) is applied to the two selected parents. Trip delimiters are then inserted by an efficient linear-time Split so as to make a

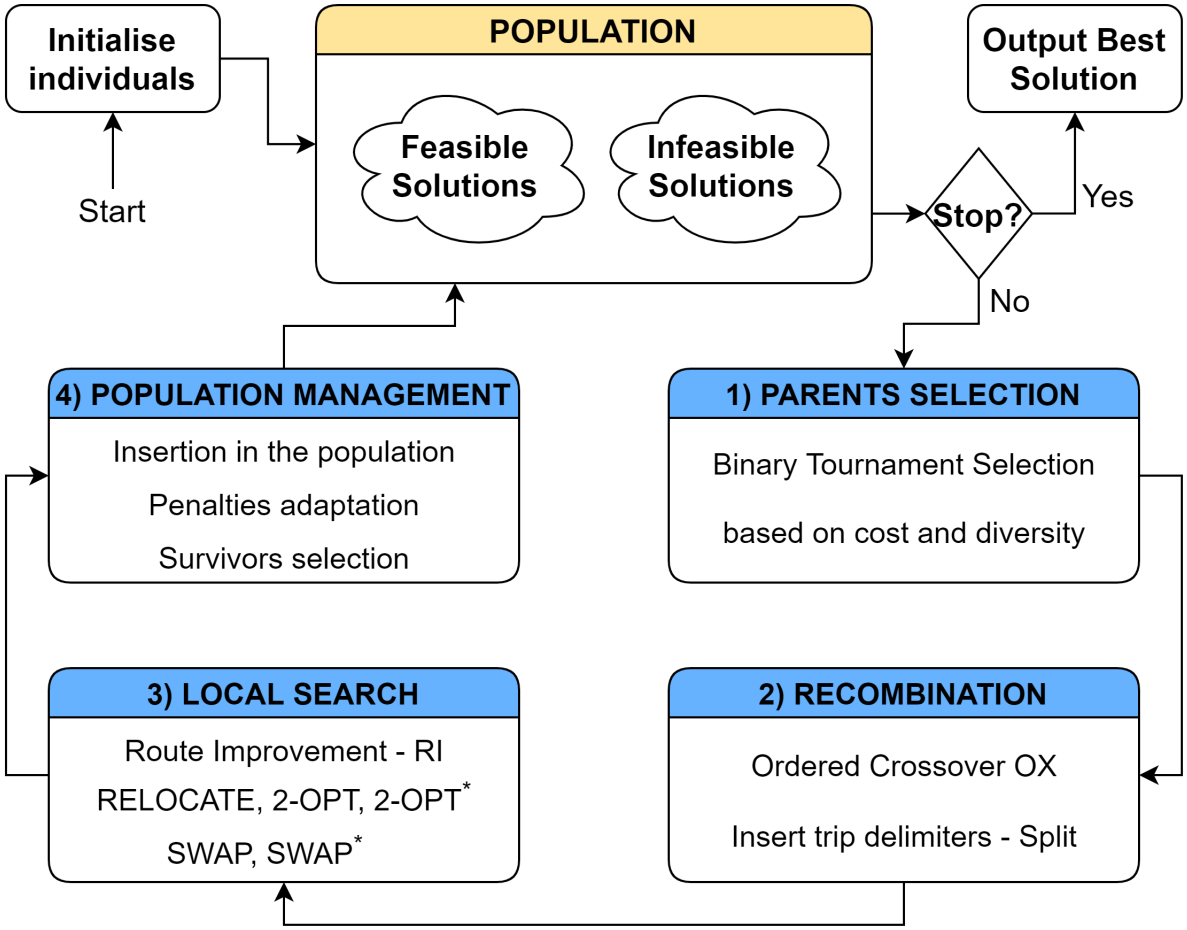

Figure 14 – General structure of HGS-CVRP (inspired by (VIDAL, 2022))

complete CVRP solution. Ordered crossover (OX - (OLIVER; SMITH; HOLLAND,)), includes 2 steps: 1) selecting a random slice of the first individual and then 2) completing missing values by starting from the second cutting point and adding the sequence from the second individual.

Step 3 – Local Search: The proposed local search is a neighborhood search including a route improvement (RI) which is a combination of classical RI (Relocate, Swap, 2-Opt and 2-Opt*) and Swap*. Further, a Repair operation with 50% probability is applied to infeasible solutions to recover feasible solutions. In the classical form of the SWAP neighborhood, 2 customers are exchanged, aiming to make any intra-route or inter-route improvements. However, SWAP* neighborhood exchanges 2 customers from 2 different routes without an insertion in place. It means that one customer from route1 can be inserted into any position of route2 and one customer from route2 can be inserted into any position of route1.

Step 4 – Population management: Each new individual produced is inserted in the appropriate subpopulation - feasible or infeasible. If a subpopulation reaches the maximum individuals, a Survivor selection is triggered. In addition, penalty parameters

for solution infeasibility are adapted in this step.

For further details, see (VIDAL, 2022).

3 LITERATURE REVIEW

The Vehicle Routing Problem (VRP) is widely recognized as one of the notable success stories in the field of operations research. The VRP encompasses various versions and complexities, with the Capacitated Vehicle Routing Problem (CVRP) being a fundamental and significant combinatorial optimization problem. The CVRP finds extensive applications in numerous real-world scenarios, including but not limited to food distribution, pharmaceutical distribution, urban bus routing, garbage collection, transportation of hazardous materials, dairy industry (TARANTILIS; KIRANOUDIS, 2007), and distribution of ready-mixed concrete (SCHMID et al., 2010).

In the context of the Capacitated Vehicle Routing Problem (CVRP), the vehicles are assumed to be identical and operate from a single central depot. The primary constraint in the CVRP is the capacity limitation of the vehicles, while other factors such as time windows or vehicle availability can be considered depending on the problem formulation. The objective of the CVRP is to minimize the overall cost associated with serving all the customers, which typically involves minimizing the total distance traveled or the number of vehicles used. To address the CVRP, two major categories of optimization algorithms are commonly employed: exact methods and heuristic algorithms (also known as approximate algorithms) (TOTH; VIGO, 2014).

3.1 Exact methods

The Capacitated Vehicle Routing Problem (CVRP) is considered an extension of the renowned Traveling Salesman Problem (TSP), which involves finding the minimum-cost Hamiltonian circuit that visits a given set of points exactly once. Consequently, many exact approaches for solving the CVRP have been developed based on the extensive and successful research conducted for exact solutions of the TSP. Toth and Vigo (TOTH; VIGO, 2014) have classified these approaches into three categories: Branch and Cut and Price, Branch and Cut, and reduced set partitioning.

In the literature, several exact methods have emerged as highly successful in solving the CVRP. These include the Branch-and-Cut algorithm (LYSGAARD; LETCHFORD; EGLESE, 2004; AUGERAT et al., 1995; LETCHFORD; EGLESE; LYSGAARD, 2002), Branch and Bound (CHRISTOFIDES; EILON, 1969), and the Branch-Cut-and-Price method, which combines the principles of Branch and Cut with Lagrangean relaxation and column generation techniques (FUKASAWA et al., 2006; BALDACCI; CHRISTOFIDES; MINGOZZI, 2008; PECIN et al., 2017).

3.2 Heuristic, Metaheuristic, and hybrid methods

Due to the NP-hard nature of the Capacitated Vehicle Routing Problem (CVRP), exact methods face significant computational challenges when solving large-scale instances. Achieving optimal solutions for CVRP instances using exact methods can be time-consuming, especially when the number of customers exceeds 200 (UCHOA et al., 2017). Therefore, researchers have turned to heuristic, metaheuristic, and hybrid approaches to obtain satisfactory solutions within a reasonable time frame.

3.2.1 Heuristic approaches

In recent years, numerous sophisticated mathematical programming decomposition algorithms have been proposed to address the Vehicle Routing Problem (VRP). Despite these advancements, optimal solutions can only be obtained for relatively small instances, typically involving around 100 customers, and the computation times exhibit significant variability. However, real-life VRP instances often involve large-scale problem sizes and necessitate prompt solutions within predictable time frames. Consequently, efficient heuristic approaches are indispensable in practical applications.

Moreover, the precise problem formulation of the VRP can vary across different settings, introducing additional challenges. Consequently, the development of flexible heuristics capable of accommodating diverse objectives and side constraints becomes essential. To address these concerns, researchers such as Toth and Vigo (TOTH; VIGO, 2014) have focused on devising heuristics that exhibit robustness and adaptability to various problem specifications.

The nearest-neighbor heuristic is one of the simplest constructive heuristics employed in the Vehicle Routing Problem (VRP). This method begins at the depot and incrementally expands a route by visiting the nearest unrouted customer, selected from those compatible with the remaining capacity of the vehicle. Once no more customers can be added, the vehicle returns to the depot, and a new route is initiated. Another constructive approach, known as the Clarke and Wright algorithm (CLARKE; WRIGHT, 1964), focuses on route mergers. Initially, each customer is visited by a dedicated route. The algorithm then evaluates potential mergers between pairs of routes and selects the merger that yields the greatest cost savings.

In addition to these methods, insertion heuristics are also widely employed, with the approach proposed by Mole and Jameson (MOLE; JAMESON, 1976) being a popular example. Insertion heuristics construct a solution through successive insertions of customers, guided by weighted insertion costs.

Two-phase methods are designed to tackle the Vehicle Routing Problem (VRP) by

reducing it to a Traveling Salesman Problem (TSP). Cluster-first, route-second approaches involve creating clusters of customers whose total demand fits within the vehicle capacity and then solving a TSP for each cluster. Gillett and Miller's heuristic (GILLETT; MILLER, 1974) provides an example of this method, where clusters are defined as angular sectors centered on the depot. Fisher and Jaikumar's method (FISHER; JAIKUMAR, 1981) addresses a generalized assignment problem during the clustering phase. Another approach, known as the petal heuristic (BALINSKI; QUANDT, 1964), constructs a large number of routes and subsequently solves a set partitioning problem to select a subset of routes that visit each customer exactly once. On the other hand, route-first, cluster-second heuristics (BEASLEY, 1983; PRINS; LABADI; REGHIOUI, 2009) adopt a different strategy. They relax the vehicle capacity constraints to solve a TSP, resulting in a route that covers all customers, often referred to as a "giant tour". A splitting procedure is then applied to divide the giant tour into capacity-feasible routes, transforming it into a valid VRP solution.

An improvement procedure, also known as local search, begins with an initial solution, denoted as s, typically obtained through a constructive heuristic. It focuses on exploring a subset $N(s)$ of solutions that are structurally close to s, referred to as the neighborhood of s. In practice, $N(s)$ is implicitly defined by a transformation $s \rightarrow s^{'}$, known as a move, rather than explicitly generating all solutions in the neighborhood. The moves have been initially defined for the Traveling Salesman Problem (TSP) and can also be applied to each route in the Capacitated Vehicle Routing Problem (CVRP). For example, one type of move is node relocation, where a customer is removed from its current position and reinserted at a different position within a route. Another move is node exchange, where the positions of two customers are swapped within a route. The neighborhoods defined by these moves can be explored in $O(n^2)$ time complexity. In addition to these basic moves, there are $k - opt$ moves (LIN; KERNIGHAN, 1973), which are more efficient. The $k - opt$ moves involve removing k edges from a route and reconnecting the resulting subsequences with k other edges, leading to potentially better solutions.

The $Or - opt$ move, proposed by Or (OR, 1976), involves relocating a contiguous sequence of 1 to λ consecutive customers within a route. On the other hand, Osman (OSMAN, 1993) introduced the $\lambda - interchange$ move, which exchanges two sequences of customers, each containing at most λ customers (the two sequences may have different lengths). These moves provide additional flexibility and exploration capabilities in local search algorithms for the Vehicle Routing Problem (VRP) and its variants.

3.2.2 Metaheuristic approaches

The local search heuristics for routing problems have evolved into metaheuristics, which comprise a range of techniques aimed at escaping local optima and achieving improved results within reasonable computation times, in comparison to exact algorithms. Given the highly combinatorial nature of vehicle routing problems, local search procedures are typically integral components of effective metaheuristics. However, there are a few exceptions, such as simulated annealing, basic versions of genetic algorithms, ant colony optimization, and particle swarm techniques, which do not explicitly incorporate local search components.

Simulated annealing is now less commonly utilized for routing problems, despite being one of the earliest metaheuristics introduced for the CVRP. Osman's work in 1993 presented $\lambda-interchange$ moves as part of simulated annealing (OSMAN, 1993). Although effective implementations of simulated annealing for routing problems can still be found occasionally, such as Lin and Yu's work on the team orienteering problem (LIN, 2013) and Lin et al.'s work on the TTRP with time windows (LIN; VINCENT, 2012). Deterministic variants of simulated annealing have been more successful in recent years. For instance, Li et al. (LI; GOLDEN; WASIL, 2005) proposed a record-to-record travel method specifically for the CVRP, which has shown promise, especially in parallel implementations (GROëR; GOLDEN; WASIL, 2011).

Variable neighborhood search (VNS) and its simpler variant, variable neighborhood descent (VND), are metaheuristics known for their speed, efficiency, and simplicity. They operate by exploring different neighborhoods of increasing sizes, with the underlying idea that a local optimum in one neighborhood may not be optimal in others. VNS and VND are frequently used to replace traditional local search algorithms within other metaheuristics. In the context of the CVRP, iterated local search methods that incorporate VND have been proposed (CHEN; HUANG; DONG, 2010), as well as for the CARP with split deliveries (BELENGUER et al., 2010). Effective VNS algorithms have been developed for various problems such as the open VRP (FLESZAR; OSMAN; HINDI, 2009), multi-depot VRP (KUO; WANG, 2012), inventory routing problem (LIU; CHEN, 2012; POPOVIć; VIDOVIć; RADIVOJEVIć, 2012), and CARP (HERTZ; MITTAZ, 2001; POLACEK et al., 2008). Although these methods may be outperformed by more advanced metaheuristics, their fast execution makes them valuable candidates for solving large-scale problems (KYTöJOKI et al., 2007).

Similar to the aforementioned metaheuristics, the greedy randomized adaptive search procedure (GRASP) (FEO; RESENDE, 1989) is not highly efficient when applied to routing problems. This inefficiency can be attributed to its independent iterations, where a solution is generated using a randomized greedy heuristic and then improved

through local search. Although Marinakis proposed a basic version of GRASP for the CVRP (MARINAKIS, 2012), more efficient variants have been developed by incorporating additional components. For instance, the path relinking technique has been integrated into GRASP for the LRP (PRINS; PRODHON; CALVO, 2006), two-echelon LRP (NGUYEN; PRINS; PRODHON, 2012), and CARP (USBERTI; FRANçA; FRANçA, 2013). In another study, Qu and Bard (QU; BARD, 2012) employed a large-neighborhood search as an improvement procedure within a GRASP framework for a pickup and delivery problem.

Iterated local search (ILS) (LOURENçO; MARTIN; STüTZLE, 2010) and guided local search (GLS) (KILBY; PROSSER; SHAW, 1999) are highly effective metaheuristics when applied to vehicle routing problems. They operate by generating a sequence of local optima through the iterative process of alternating between a local search and a perturbation step. In the case of ILS, the solution is directly perturbed, while GLS perturbs the edge costs, causing the local optimum to no longer be optimal under the modified costs. Notable examples of their effectiveness include an ILS for the heterogeneous fleet VRP (SUBRAMANIAN et al., 2012) and a GLS for the CARP (BEULLENS et al., 2003).

In the 1990s, tabu search methods emerged as highly effective metaheuristics for solving vehicle routing problems. To minimize a penalized objective function, vehicle capacity and time windows were frequently relaxed. Notable achievements were made in various problem domains, such as the Capacitated Vehicle Routing Problem (CVRP) (BARBAROSOGLU; OZGUR, 1999; REGO; ROUCAIROL, 1996; TOTH; VIGO, 2003), the Vehicle Routing Problem with Time Windows (VRPTW) (CORDEAU; LAPORTE; MERCIER, 2001), the Heterogeneous Fleet VRP (HFVRP) (BRANDãO, 2011), and the CVRP and HFVRP with two-dimensional loading (LEUNG et al., 2013; LEUNG et al., 2011). While these algorithms often employed classical moves, some innovations were introduced. For example, Rego and Roucairol (REGO; ROUCAIROL, 1996) implemented ejection chains, and Toth and Vigo (TOTH; VIGO, 2003) devised a granular tabu search (GTS) that restricts moves to a small fraction of the edges (specifically, the cheapest ones), which is dynamically adjusted throughout the algorithm.

Genetic algorithms (GAs) were introduced shortly after the initial metaheuristics for vehicle routing, namely simulated annealing and tabu search. However, their performance has been mixed, except in the case of the Vehicle Routing Problem with Time Windows (VRPTW) (THANGIAH, 2019; POTVIN; BENGIO, 1996). In the study presented in (THANGIAH, 2019), chromosomes were used to represent complete solutions, where each chromosome comprised a list of customers assigned to successive routes, separated by copies of the depot node. Crossover operations based on this encoding, such as RBX (POTVIN; BENGIO, 1996), could generate offspring solutions with route violations in terms of vehicle capacity. Although this problem could be resolved by relocating customers to other routes, it led to a degradation in the genetic transmission of beneficial patterns

from parents to children. Another factor contributing to these somewhat unsatisfactory results was the absence of a local search component.

Significant progress in solving the Capacitated Vehicle Routing Problem (CVRP) was achieved from 2003 onwards with the introduction of memetic algorithms (MAs), which combine genetic algorithms with local search techniques applied to offspring solutions with a certain probability. The pioneering work by Berger and Barkaoui (BERGER; BARKAOUI, 2003) laid the foundation, although they still used complete solutions as chromosomes. The challenge of capacity violations was addressed by Baker and Ayechew (BAKER; AYECHEW, 2003). In their approach, each chromosome represents a partition of customers into clusters. The decoding process involves solving a Traveling Salesman Problem (TSP) for each cluster using a constructive heuristic followed by a local search utilizing $2-opt$ and $\lambda-interchange$ moves. Prins (PRINS, 2004) pursued a different strategy known as the route-first cluster-second approach, where vehicle capacities were relaxed, enabling the use of chromosomes without route delimiters, similar to those used in the TSP. Prins referred to these chromosomes as giant tours. A procedure called Split was developed to derive an optimal solution to the CVRP from each chromosome, subject to the prescribed sequence. Notably, classical TSP crossovers such as LOX and OX could be readily applied in this context, providing an advantage in terms of algorithmic design and reuse.

The memetic algorithm (MA) introduced by Prins marked a significant breakthrough, surpassing the performance of tabu search methods. Following this success, several other effective memetic algorithms were developed, employing the concept of giant tours, for various vehicle routing problems. These include the Capacitated Arc Routing Problem (CARP) (LACOMME; PRINS; RAMDANE-CHERIF, 2004), the multiperiod Location Routing Problem (LRP) (PRODHON; PRINS,), the multicompartment Vehicle Routing Problem (VRP) (FALLAHI; PRINS; CALVO, 2008), a production-distribution problem (BOUDIA; PRINS, 2009), and the cumulative CVRP (NGUEVEU; PRINS; CALVO, 2010). Notably, Nagata and Bräysy (NAGATA; BRäYSY, 2009) proposed an innovative MA for the CVRP that did not rely on giant tours. Their approach utilized a sophisticated crossover operator known as edge assembly crossover, which demonstrated remarkable effectiveness in solving the CVRP.

Ant colony optimization (ACO) is particularly suitable for problems that involve constructing solutions by finding paths in a graph. In the context of routing problems, ants can be employed to create routes by selecting arcs in a sequential manner, guided by pheromone deposits and a nearest neighbor heuristic. Recognizing the potential of this approach, Reimann et al. (REIMANN; DOERNER; HARTL, 2004) devised a clever strategy where ants build routes through successive insertions, starting from loops around the depot. By incorporating a local search procedure, their algorithm yields highly effective results

for the Capacitated Vehicle Routing Problem (CVRP). Another noteworthy contribution in the field of ACO algorithms is the work of Santos et al. (SANTOS; COUTINHO-RODRIGUES; CURRENT, 2010), who developed an ACO algorithm equipped with local search. This algorithm stands out as one of the top-performing metaheuristics for the Capacitated Arc Routing Problem (CARP).

Particle swarm optimization (PSO) has gained attention in the field of vehicle routing relatively recently, and its effectiveness has been demonstrated primarily through hybrid approaches that combine it with other metaheuristics. In their work, Chen et al. (KENNEDY; EBERHART,) proposed a PSO algorithm for the Capacitated Vehicle Routing Problem (CVRP) that focuses on assigning customers to vehicles, while the actual routes are determined using a simulated annealing step applied to each vehicle. Marinakis and Marinaki (MARINAKIS; MARINAKI, 2010) achieved even better results by employing a more complex hybridization strategy that combines PSO with GRASP and path relinking. Furthermore, two PSO algorithms specifically tailored for the CVRP with stochastic demands have been developed (MARINAKIS; IORDANIDOU; MARINAKI, 2013; MOGHADDAM; RUIZ; SADJADI, 2012). These approaches reflect the ongoing exploration of PSO's potential for solving vehicle routing problems.

Imperialist Competitive Algorithm (ICA) (ATASHPAZ-GARGARI; LUCAS,) is a social inspired evolutionary algorithm, achieving success in many applications areas: image encryption (ENAYATIFAR; ABDULLAH; LEE, 2013), stock market forecasting (SADAEI et al., 2016) and scheduling problem (AYOUGH; ZANDIEH; FARSIJANI, 2012). Yousefikhoshbakht et al. (YOUSEFIKHOSHBAKHT; SEDIGHPOUR, 2013) solved the Traveling Salesman Problem based on a refined ICA, applying order crossover in the Assimilation step and two exchange moves in Revolution step.

3.2.3 Hybrid methods

Another direction in the field of solving the CVRP is the development of hybrid methods that combine multiple components to enhance performance. It has been observed that the most effective metaheuristics for the CVRP, such as memetic algorithms and Evolutionary Local Search (ELS), incorporate a local search procedure (VIDAL et al., 2012; NAGATA; BRäYSY, 2009; PRINS; SEVAUX; SÖRENSEN, 2004). In some cases, this local search procedure is replaced with techniques like Variable Neighborhood Descent (VND), Variable Neighborhood Search (VNS), or Large Neighborhood Search (LNS) to enhance intensification.

However, many existing metaheuristics are limited to solving a single variant of the routing problem, which hinders their integration into commercial software. The development of methods capable of addressing multiple problem variants with a single algorithm is

still in its early stages. Some notable examples of such methods include the Universal Tabu Search Algorithm (UTSA) proposed by Cordeau et al. (CORDEAU; LAPORTE; MERCIER, 2001), the Large Neighborhood Search (LNS) approach introduced by Pisinger and Röpke (PISINGER; ROPKE, 2007), and the Hybrid Genetic Algorithm proposed by Vidal et al. (VIDAL et al., 2012) for solving CVRP, Multi-Depot VRP (MDVRP), and Periodic VRP (PVRP). Vidal et al. also proposed another GA-based approach for problems with time windows (VIDAL et al., 2013). The Unified Hybrid Genetic Search (UHGA) combines evolutionary search, local search, and population management to address the MDVRP, PVRP, and MDPVRP. Sbai et al. proposed two metaheuristics for solving the CVRP in a real-life case study, utilizing a GA combined with a specialized Variable Neighborhood Search (VNS) (SBAI; KRICHEN; LIMAM, 2022). More recently, Vidal proposed a Hybrid Genetic Search for CVRP (VIDAL, 2022), known as HGS-CVRP. The overall methodology of HGS-CVRP is similar to UHGA, but it introduces an additional neighborhood called SWAP* Neighborhood, which allows exchanging two customers from different routes without requiring an insertion in place.

4 ICAHGS APPROACH

In this study, a novel algorithm is proposed which aims to incorporate the benefits of both ICA and HGS-CVRP approaches. Additionally, the algorithm incorporates various modifications to both ICA and HGS-CVRP to further enhance its performance. Prior to the detailed description of the proposed algorithm, the applied refinements will be discussed in the following subsections.

4.1 Refined ICA

In order to enhance the adaptability and suitability of the ICA method for addressing the CVRP problem, refinements have been applied to the original ICA.

4.1.1 Modified "Create initial empires"

Following the original ICA (see section 2.6), initialization of countries and their assignment to colonies are based on randomly generated individuals. For CVRP such individuals may be either feasible or infeasible solutions. Given that imperialists play "*Local Optima*" roles in their corresponding empires, initialization needs to ensure that the solutions assigned are in fact feasible solutions i.e. N_{imp} feasible solutions are generated.

Equations (4.1) and (4.2) are applied to calculate the number of colonies for each empire based on their power (or fitness value). These two equations form a normalization equation to ensure that more colonies belong to more powerful empires. It should be noted that the value "1.3" is not part of the original ICA, However, it appears in the first implementation of the algorithm in MATLAB (MATLAB. . . ,). Finally, the Total cost of each empire is calculated, as before, by Equation (2.3).

$$maxPower = 1.3 \times MAX\Big(Cost\Big(imperialist_1\Big), \ldots, Cost\Big(imperialist_{N_{imp}}\Big)\Big) \tag{4.1}$$

$$N.C._n = round\left(\frac{maxPower - Cost\big(imperialist_n\big)}{\sum_{i=1}^{N_{imp}}\Big(maxPower - Cost\big(imperialist_i\big)\Big) \times N_{col}}\right) \tag{4.2}$$

where $N.C._n$ is the number of colonies of the n^{th} empire, N_{col} is the total number of colonies and the $Cost$ represents the fitness function (or objective function). Algorithm 2 represents the pseudo code for this step.

Algorithm 2 Create initial empires

1: Genearte N_{imp} random **feasible solutions**
2: Calculate the number of colonies for each empire by Equations (4.1) and (4.2)
3: Generate $N.C.$ random solutions (*Feasible* or *Infeasible*) for each empire
4: Calculate the Total cost of each empire using Equation (2.3)

Algorithm 3 AssimRevol (combined Assimilation and Revolution)

1: **for** each empire **do**
2: Calculate number of colonies to participate in *AssimRevol* by Equation (2.2) (the minimum number of participated colonies is one)
3: **for** $i = 1$ to *NumOfRevolvedColonies* **do**
4: **if** $AssimType = False$ **then**
5: do *Crossover* between random selected colonies
6: **else**
7: do *Crossover* between *BestSolutionOverall* and random selected colonies
8: **end if**
9: *LocalSearch* as *Revolution* operator by HGS-CVRP
10: *Population management* and *Penalty management* by HGS-CVRP
11: **if** *BestSolutionOverall* is not improved after *nbIter* iterations **then**
12: run *Restartmechanism*
13: **end if**
14: **end for**
15: $AssimType = False$
16: **end for**

4.1.2 Defining "AssimRevol" as a new step

The "*Assimilation*" and "*Revolution*" steps of the original ICA are combined and refined to form a new step, "*AssimRevol*" as described in Algorithm 3. The crossover between the imperialist and a colony (see section 2.6), is adjusted to be performed between two random colonies from the same empire - selected by two Binary Tournament Selections. Such increased randomness in the colony selection supports diversification – see section 2.5; avoiding premature convergence of the empires and thus aiming to have a positive effect on the quality of solutions achieved.

Further, when triggered by multi-step restart mechanism (by setting $AssimType = True$) - see section 4.2.1; crossover is applied to the Best Solution Overall and one randomly selected colony so as to increase intensification – see section 2.5.

As indicated in line 1 of Algorithm 3, *AssimRevol* is performed for each empire. The number of colonies to participate in the *AssimRevol* is calculated by Equation (2.2) – see line 2. In lines 3 to 14 the adjustments to the selected colonies are described. First, crossover is performed (lines 5 and 7) either between randomly selected colonies or between the *BestSolutionOverall*" and randomly selected colonies, controlled by the parameter

AssimType, line 4. Each colony generated from the crossover operator is passed to the *LocalSearch* function, HGS-CVRP (lines 9 and 10). *Revolution* (mutation), population management and penalty management are applied. In line 12, the *RestartMechanism* is triggered in the case of no improvement in the *BestSolutionOverall* after *nbIter* iterations.

4.1.3 Removing the step "Possess Empires"

"*Possess Empires*" is step 4 in the original ICA where a colony and its imperialist are exchanged when the colony has a better fitness value (see section 2.6). To avoid this step, a simple representation change is needed where the imperialist and its colonies are a sorted list, with the fittest colony being the imperialist.

4.1.4 Modified "Imperialistic Competition"

Randomized selection is applied to refine step 6 in the original ICA (see section 2.6) to increase diversification (see section 2.5). A random colony (rather than the weakest) is selected from the weakest empire in addition to a randomly selected empire (rather than the most powerful one). Further, when the weakest empire loses all its colonies, its imperialist will be seized by a randomly selected empire (rather than the most powerful one).

4.2 Refined HGS-CVRP

In order to enhance the performance of the HGS-CVRP method, the single-step restart mechanism is replaced with a proposed multi-step restart mechanism.

4.2.1 Proposed multi-step restart mechanism

HGS-CVRP involves a complete restart after a predefined number of generations given no improvement in the *BestSolutionOverall*. Such a complete restart requires time to regain the fitness level at the point of restart and thus, a need for a more efficient solution was recognized.

The *multi-step restart mechanism* proposed herein provides two types: ***Partially*** and ***Totally*** (original). As shown in Figure 15, in *RestartPartially*, 25% of the colonies (orange) will be kept and 75% of the colonies (blue) will be regenerated randomly. Further, 25% of the kept colonies (orange) will be selected randomly from *Feasible* solutions and 75% from *Infeasible* ones. Infeasible solutions have more potential to become new Feasible solutions, while most of the Feasible solutions, at this stage, have similar fitness values. As described in Algorithm 4 (lines 1-8), the *number of resets* and *last reset value* will be

stored. The parameter *number of resets* defines whether *RestartPartially* will be performed (line 10) or *RestartTotally* (lines 11 to 17). Lines 13-14 select how crossover should be applied.

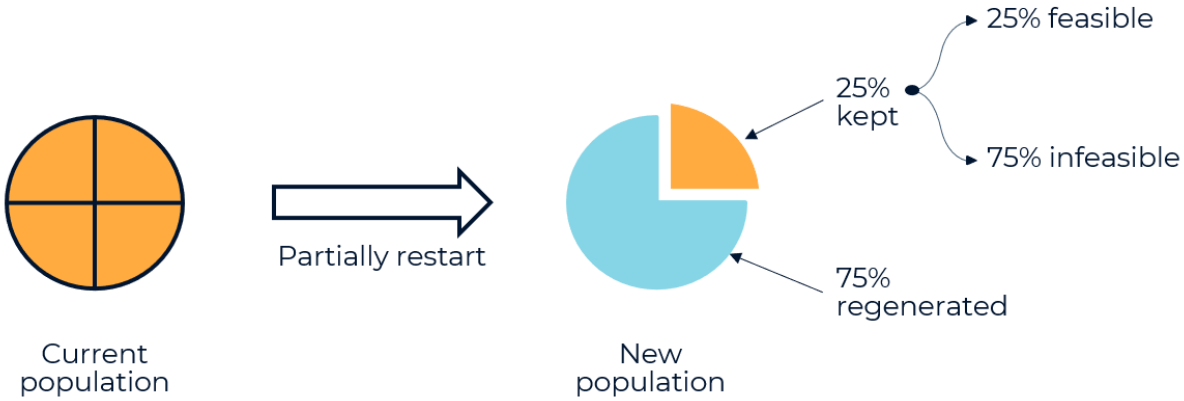

Figure 15 – The process of partially restart

Algorithm 4 The proposed multi-step restart mechanism

1: Define variable *NumOfResets* to track the number of resets for each empire
2: Define variable *LastResetValue* to keep the last imperialist's fitness value when *Restart* mechanism is started
3: **if** $imperialistCost = LastResetValue$ **then**
4: $NumOfResets = NumOfResets + 1$
5: **else**
6: $LastResetValue = imperialistCost$
7: $NumOfResets = 1$
8: **end if**
9: **if** $NumOfResets = 1$ **then**
10: run *RestartPartially*
11: **else if** $NumOfResets = 2$ **then**
12: run *RestartTotally*
13: **if** $imperialistCost \neq BestSolutionCost$ **then**
14: $AssimType = True$
15: **end if**
16: **else if** $NumOfResets > 2$ **then**
17: run *RestartTotally*
18: **end if**

4.3 ICAHGS approach

The proposed ICAHGS, seeks to exploit the advantages of both HGS-CVRP and ICA in a single method. As stated, ICA shows strength in global search in its multi-population structure whilst HGS-CVRP benefits from its population management, diversity control and improved local search.

Figure 16 provides an overview of the proposed ICAHGS algorithm. As shown, ICA is the main algorithm – ICA Workspace; based on a refined ICA – see section 4.1. Further, within each empire of the ICA Workspace, a refined HGS-CVRP manages the colonies – see section 4.2. Algorithm 5 presents the ICAHGS algorithm, where the main algorithm is the refined ICA and lines 7 to 12 apply the refined HGS-CVRP to the colonies.

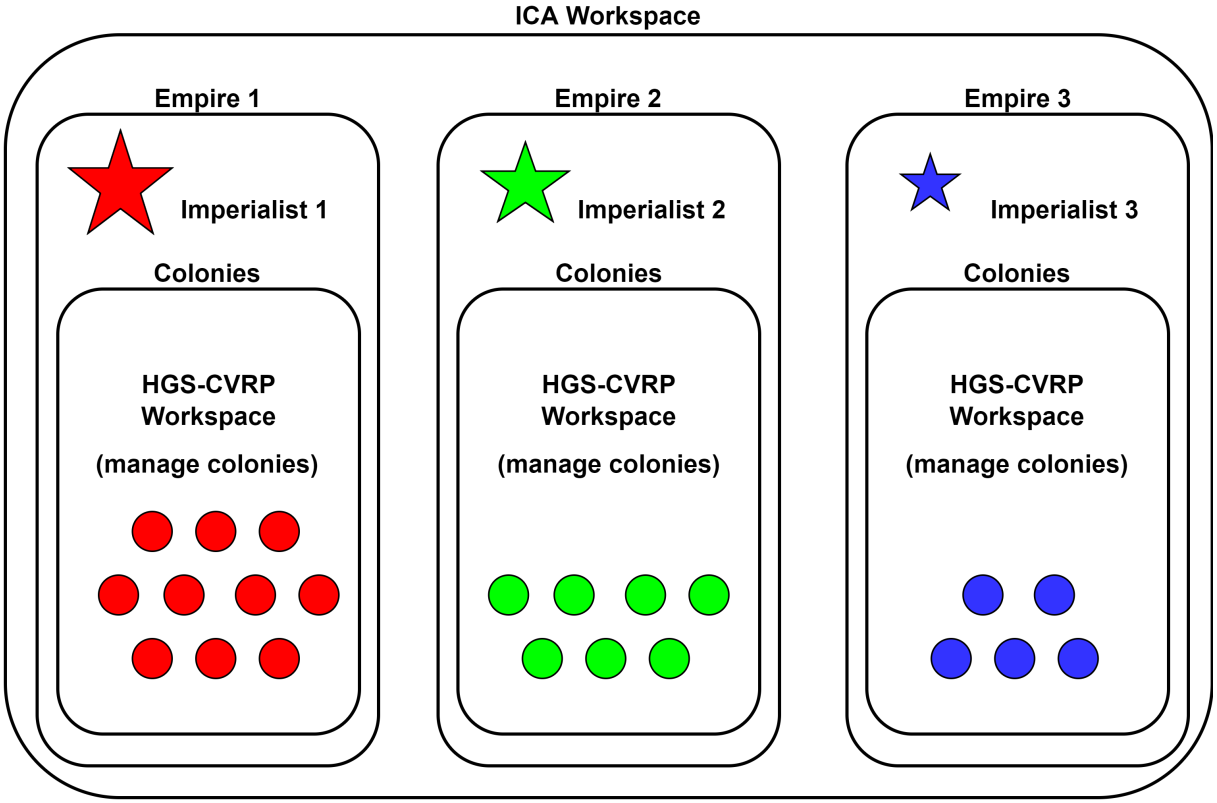

Figure 16 – An overview of the proposed algorithm, ICAHGS

4.4 Experimental results

4.4.1 Experimental setup

The simulator was developed in C++ and compiled with g++ 7.5.0 on Linux Ubuntu 18.04. All experiments were performed on an Intel(R) Core (TM) i5-4590T CPU @ 2.00GHz with 12 GB RAM. To enable comparison of experimental results with Vidal's (VIDAL et al., 2012), the T_{max} applied is scaled up by $R = 2186/1646$ where R is the ratio of the single thread rating of CPUs (CPU...,). Therefore, the new time limit will be $T_{max} = T_{maxV} \times R$ seconds, where n is the number of customers and $T_{maxV} = n \times 240/100$ is the suggested T_{max} by Vidal.

To increase statistical significance, 10 independent runs with different seed numbers are performed for each instance. Further, to measure the quality of solutions, the Gap of each algorithm was calculated by $Gap = (Value - BKS)/BKS$ where $Value$ is the solution

Algorithm 5 ICAHGS

```
Start
Create initial empires (Algorithm 2)
while stopping criterion is not met do
   AssimRevol step:
   for each empire in empires do
      for each colony in Revolving_Colonies do
         Crossover OX
         Local Search
         Population management & penalties adaptation
         if no improvement after nbIter iterations then
            run Restart Mechanism
         end if
      end for
   end for
   Imperialistic competition - The gradual destruction of weaker empires
end while
Output the best-found solution
End
```

value provided by the proposed methods and BKS is the Best-Known Solution value for instance, extracted from the CVRPLIB website (CVRPLIB. . . ,). Further, statistical tests - t-tests and the Wilcoxon test, were applied where possible, provided in section 4.4.9.

4.4.2 Benchmarks

In this study, 140 benchmark instances, selected from the following benchmarks are applied:

- **Uchoa (UCHOA et al., 2017):** This benchmark contains 100 instances from 100 to 1000 clients providing a wide range of CVRP characteristics. In this study, the main experiments are performed on this benchmark, and are referred to X-instances.
- **CMT (CHRISTOFIDES, 1979):** CMT is another well-known benchmark in the literature, containing 14 instances from 50 to 199 customers.
- **Golden (GOLDEN et al., 1998):** Golden is composed of 20 large-scale instances using from 200 to 480 customers where some instances have constraints on the maximum length of any route.
- **LoggiBUD (LOGGI. . . ,):** The Loggi Benchmark for Urban Deliveries provides a real-world dataset - simulating the challenges of a large delivery company in the last-mile step of its supply chain in Brazil's largest cities. The 6 instances of the DIMACS VRP challenge 2021 (DIMACS,) were selected.

4.4.3 Tuning the ICA parameters

Two subsets of the parameters "Initial Imperialists" ($subset_1 = [2, 3, 5]$) and "Revolution Rate" ($subset_2 = [0.4, 0.5, 0.6]$) were experimentally defined. 5 random instances from different client ranges of X-instances were selected and the ICAHGS was performed for 10 independent runs with different seed numbers. The ξ (zeta) from the original ICA and the HGS-CVRP parameters were kept unchanged from (MATLAB. . . ,) and (VIDAL, 2022), respectively. The average performance (Value) and average efficiency (time) – in term of consumed CPU time in seconds; are presented in Table 1. The resulting best values for "Initial Imperialists" (II) and "Revolution Rate" (RR) are 3 and 0.5, respectively. The overview of the ICAHGS parameters is provided in Table 2.

Table 1 – Sweep of ICA Parameter

Cases		X-n129-k18		X-n228-k23		X-n336-k84		X-n524-k153		X-n801-k40	
II	**RR**	**Value**	**time**	**Value**	**time**	**Value**	**time**	**Value**	**time**	**Value**	**time**
2	0.4	28940.0	10.20	25742.8	41.76	139269.7	637.02	154669.3	997.12	73431.3	2302.63
2	0.5	28940.0	10.20	25742.8	31.13	139193.7	841.64	154674.7	681.63	73459.6	2450.21
2	0.6	28940.0	9.65	25742.7	83.13	139246.3	959.12	154719.0	1128.96	73467.7	2049.22
3	0.4	28940.0	14.83	25742.7	165.01	139293.7	898.18	154689.0	1365.86	73547.0	1890.11
3	0.5	28940.0	13.35	25742.0	212.75	139186.3	787.86	154666.4	927.58	73414.3	1978.21
3	0.6	28940.0	18.03	25742.7	97.48	139275.7	381.11	154726.3	957.84	73586.0	2092.21
5	0.4	28940.0	24.72	25743.0	59.40	139385.0	670.66	154683.7	948.48	73432.5	2136.76
5	0.5	28940.0	24.92	25742.8	111.02	139270.7	733.09	154689.7	1149.70	73437.0	1997.55
5	0.6	28940.0	32.03	25742.7	167.63	139316.7	414.46	154746.8	889.11	73423.0	2526.59

Table 2 – The parameters of ICAHGS

	Parameter name	Additional comment	Value
ICA part	Initial countries	From the original HGS-CVRP	100
	Initial imperialists	From parameter tuning	3
	Revolution Rate	From parameter tuning	0.5
	ξ (zeta)	From the original ICA	0.3
HGS-CVRP part	μ (miu)	Minimum population size	25
	λ (lambda)	Generation size	40
	n_{elite}	Number of elite solutions considered in the fitness calculation	4
	$n_{closest}$	Number of close solutions considered in the diversity-contribution measure	5
	Γ	Granular search parameter	20
	ξ^{Ref}	Target proportion of feasible individuals for penalty adaptation	0.2

4.4.4 Refined ICA vs. original ICA

In order to assess the performance and efficiency of the proposed refined ICA within ICAHGS, two approaches are considered: ICAHGS with the original ICA (see section 2.6) and with the refined ICA (see section 4.1).

The comparison is performed on 10 random benchmark instances and the results are listed in Table 3. In the table, if BKS is reached, BKS is listed as the improvement; otherwise, the % improvement is listed either with respect to the value achieved or to the gap. If both versions achieved BKS i.e., no improvement is needed, the effect on time is given.

The ICAHGS with refined ICA found 5 BKS. In 4 instances, both algorithms achieved the BKS, but ICAHGS with refined ICA showed better efficiency. Moreover, for X-n856-k95, the performance of ICAHGS with respect to the gap, has improved by more than 50%.

Table 3 – Comparative results to analyze the refined ICA

Instances	ICAHGS with original ICA			ICAHGS with refined ICA			improvement	
	Best	Gap	time	Best	Gap	time	type	percent
X-n106-k14	26362	0.00	162.45	26362	0.00	158.10	Time	2.68
X-n125-k30	55539	0.00	241.02	55539	0.00	20.70	Time	91.41
X-n261-k13	26558	0.00	537.42	26558	0.00	336.20	Time	37.44
X-n284-k15	20244	0.14	597.25	20215	0.00	853.60	BKS	-
X-n317-k53	78361	0.01	468.53	78355	0.00	153.33	BKS	-
X-n411-k19	19739	0.14	857.05	19712	0.00	1100.40	BKS	-
X-n429-k61	65458	0.01	1303.23	65449	0.00	571.20	BKS	-
X-n459-k26	24145	0.02	811.56	24139	0.00	1358.11	BKS	-
X-n513-k21	24201	0.00	864.97	24201	0.00	128.30	Time	85.17
X-n856-k95	89016	0.06	2584.16	88990	0.03	2150.60	Gap	50.98

4.4.5 Analysis of multi-step restart mechanism (MSRM)

To study the effect of the multi-step restart mechanism (see section 4.2.1), the performance and efficiency of ICAHGS without MSRM – allows total restart; and with MSRM – allows both partial and total restart; were compared. The results for the same 10 random instances as in section 4.4.4, are reported in Table 4. In 5 instances, MSRM has pushed the algorithm to find BKS. Furthermore, for X-n856-k95, a 40% performance improvement is achieved. For 3 cases, where BKS was achieved in both approaches, the efficiency has improved when applying MSRM.

4.4.6 ICAHGS versus HGS-CVRP on CMT and Golden benchmarks

To study the generality of the results, two further benchmarks are studied: CMT (CHRISTOFIDES, 1979) and Golden (GOLDEN et al., 1998). The average and the best

Table 4 – Comparative results to analyze the multi-step restart mechanism

Instances	ICAHGS (without MSRM)			ICAHGS (with MSRM)			improvement	
	Best	Gap	time	Best	Gap	time	type	percent
X-n106-k14	26386	0.09	58.10	26362	0.00	158.10	BKS	-
X-n125-k30	55539	0.00	21.20	55539	0.00	20.70	Time	2.40
X-n261-k13	26558	0.00	800.40	26558	0.00	336.20	Time	58.00
X-n284-k15	20244	0.14	181.10	20215	0.00	853.60	BKS	-
X-n317-k53	78355	0.00	251.80	78355	0.00	153.33	Time	39.10
X-n411-k19	19718	0.03	277.42	19712	0.00	1100.40	BKS	-
X-n429-k61	65487	0.06	818.20	65449	0.00	571.20	BKS	-
X-n459-k26	24143	0.02	1464.40	24139	0.00	1358.11	BKS	-
X-n513-k21	24201	0.00	127.23	24201	0.00	128.30	Time	-0.80
X-n856-k95	89006	0.05	408.16	88990	0.03	2150.60	Gap	40.00

obtained results at T_{max} from 10 independent runs are provided in the Table 6 (CMT) and Table 7 (Golden). In Table 5, a summary of the gaps in the average values of solutions among all instances of CMT and Golden benchmarks is presented, along with the number of BKS instances discovered.

Both ICAHGS and HGS-CVRP found 13 BKS out of 14 instances of CMT and the gap for the remaining instance (CMT5) is 0.01% - negligible for most applications.

For the golden dataset, ICAHGS achieved 8 BKS, compared to the 7 of HGS-CVRP. Furthermore, considering Average and Best found solutions, ICAHGS performed better for 10 and 8 instances, respectively – see Table 7. In addition, ICAHGS obtained 62% improvement for the Max. Gap of Best solutions and 39.1% for the Max. Gap of Average solutions.

Table 5 – Comparative results between HGS-CVRP and ICAHGS on CMT and Golden benchmarks

	CMT		Golden	
	HGS-CVRP	ICAHGS	HGS-CVRP	ICAHGS
Min Gap	0.00	0.00	0.00	0.00
Avg Gap	0.0009	0.0009	0.18	**0.14**
Max Gap	0.0123	0.0123	0.92	**0.56**
Number of BKS	13	13	7	**8**

Table 6 – Comparative results HGS-CVRP, ICAHGS and BKS for CMT benchmark

Instance	HGS-CVRP (2020)				ICAHGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
CMT1	**524.61**	0.00	**524.61**	0.00	**524.61**	0.00	**524.61**	0.00	524.61
CMT2	**835.26**	0.00	**835.26**	0.00	**835.26**	0.00	**835.26**	0.00	835.26
CMT3	**826.14**	0.00	**826.14**	0.00	**826.14**	0.00	**826.14**	0.00	826.14
CMT4	**1028.42**	0.00	**1028.42**	0.00	**1028.42**	0.00	**1028.42**	0.00	1028.42
CMT5	1291.45	0.01	1291.45	0.01	1291.45	0.01	1291.45	0.01	1291.29
CMT6	**555.43**	0.00	**555.43**	0.00	**555.43**	0.00	**555.43**	0.00	555.43
CMT7	**909.68**	0.00	**909.68**	0.00	**909.68**	0.00	**909.68**	0.00	909.68
CMT8	**865.95**	0.00	**865.95**	0.00	**865.95**	0.00	**865.95**	0.00	865.95
CMT9	**1162.55**	0.00	**1162.55**	0.00	**1162.55**	0.00	**1162.55**	0.00	1162.55
CMT10	**1395.85**	0.00	**1395.85**	0.00	**1395.85**	0.00	**1395.85**	0.00	1395.85
CMT11	**1042.12**	0.00	**1042.12**	0.00	**1042.12**	0.00	**1042.12**	0.00	1042.12
CMT12	**819.56**	0.00	**819.56**	0.00	**819.56**	0.00	**819.56**	0.00	819.56
CMT13	**1541.14**	0.00	**1541.14**	0.00	**1541.14**	0.00	**1541.14**	0.00	1541.14
CMT14	**866.37**	0.00	**866.37**	0.00	**866.37**	0.00	**866.37**	0.00	866.37

Table 7 – Comparative results HGS-CVRP, ICAHGS and BKS for Golden benchmark

Instance	HGS-CVRP (2020)				ICAHGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
Golden_1	**5623.47**	0.00	**5623.47**	0.00	**5623.47**	0.00	**5623.47**	0.00	5623.47
Golden_2	8436.24	0.38	8412.83	0.10	8417.20	0.15	8406.33	0.02	8404.61
Golden_3	11036.20	0.35	11036.20	0.35	11036.20	0.35	11036.20	0.35	10997.80
Golden_4	13624.50	0.26	13624.50	0.26	13624.50	0.26	13624.50	0.26	13588.60
Golden_5	**6460.98**	0.00	**6460.98**	0.00	**6460.98**	0.00	**6460.98**	0.00	6460.98
Golden_6	8412.90	0.15	8412.90	0.15	8412.90	0.15	8412.90	0.15	8400.33
Golden_7	10195.60	0.92	10195.60	0.92	10159.55	0.56	10132.90	0.30	10102.70
Golden_8	**11635.30**	0.00	**11635.30**	0.00	**11635.30**	0.00	**11635.30**	0.00	11635.30
Golden_9	580.39	0.12	579.71	0.00	580.30	0.10	579.71	0.00	579.70
Golden_10	737.83	0.33	736.60	0.16	737.39	0.27	736.00	0.08	735.43
Golden_11	913.80	0.20	913.14	0.13	913.60	0.18	913.06	0.12	911.98
Golden_12	1105.86	0.42	1104.95	0.34	1104.74	0.32	1103.34	0.19	1101.24
Golden_13	**857.19**	0.00	**857.19**	0.00	**857.19**	0.00	**857.19**	0.00	857.19
Golden_14	**1080.55**	0.00	**1080.55**	0.00	**1080.55**	0.00	**1080.55**	0.00	1080.55
Golden_15	1339.47	0.16	1339.07	0.13	1339.36	0.16	1337.84	0.04	1337.27
Golden_16	1614.80	0.22	1613.25	0.12	1614.50	0.20	1612.57	0.08	1611.28
Golden_17	**707.76**	0.00	**707.76**	0.00	**707.76**	0.00	**707.76**	0.00	707.76
Golden_18	**995.13**	0.00	**995.13**	0.00	**995.13**	0.00	**995.13**	0.00	995.13
Golden_19	1365.90	0.02	1365.63	0.00	1365.65	0.00	**1365.60**	0.00	1365.60
Golden_20	1818.68	0.06	1817.89	0.02	1818.37	0.04	1817.89	0.02	1817.59

4.4.7 Evaluating ICAHGS against state-of-the-art algorithms on Uchoa benchmark

ICAHGS is compared to four state-of-the-art methods from the literature: Google OR-Tools (GOOGLE...,), KGLS (knowledge guided local search) (ARNOLD; SöRENSEN, 2019), SISR (slack induction by string removals) (CHRISTIAENS; BERGHE, 2020), and HGS-CVRP (VIDAL, 2022). Google OR-Tools was selected as a baseline due to its reputation as a reliable general-purpose tool in the field of operations research. The remaining algorithms were chosen for their relevance and timeliness as representative examples of current techniques in the field.

ICAHGS experiments were conducted on 100 benchmark instances from the Uchoa benchmark. The average and best results reported in Vidal' research (VIDAL, 2022), for each state-of-the-art algorithm on X-instances, are provided for comparison.

Table 8 presents a detailed comparison of the obtained gaps in average solution values for HGS-CVRP and ICAHGS whilst Table 9 summarizes the gaps in average solution values across all instances, as well as the number of BKS instances found. A more comprehensive presentation of the comparative results can be found in Table 10.

It is clear that ICAHGS achieved higher quality solutions in comparison with its original algorithm (HGS-CVRP) and the remaining state-of-the-art algorithms. The following improvements, compared to HGS-CVRP, may be noted:

- ICAHGS achieves 51 BKS, whilst HGS-CVRP achieves 44 i.e. a 15.9% comparison (see Table 9).
- ICAHGS gains 14.3% and 27.3% improvement respectively for Max. and Avg. Gap of Average solutions (see Table 8).
- In the case of small instances (first 50 instances), the ICAHGS obtains optimal or near-optimal solutions, with 31.3% and 47.8% improvement over HGS-CVRP respectively for Max. and Avg. Gap of Average solutions (see Table 8).
- In the case of large instances (last 50 instances), the ICAHGS achieves 14.3% improvement for the Max. Gap and 23.7% for the Avg. Gap of average solutions (Table 8)

Table 8 – Comparing Gaps of average solution values for HGS-CVRP and ICAHGS

Gaps	**included instances**	**HGS-CVRP**	**ICAHGS**	**improvement (%)**
Max Gap	100	0.56	0.48	14.3
Avg. Gap	100	0.11	0.08	27.3
Max Gap	First 50	0.16	0.11	31.3
Avg. Gap	First 50	0.023	0.012	47.8
Max Gap	Last 50	0.56	0.48	14.3
Avg. Gap	Last 50	0.198	0.151	23.7

Table 9 – Comparative results between different algorithms

	OR-Tools (2020)	**KGLS (2018)**	**SISR (2020)**	**HGS-CVRP (2020)**	**ICAHGS**
Min Gap	0.38	**0**	**0**	**0**	**0**
Avg Gap	4.01	0.53	0.19	0.11	**0.08**
Max Gap	8.62	1.24	1.63	0.56	**0.48**
Number of BKS	0	6	21	44	**51**

When comparing the HGS based algorithms to other state of the art algorithms, the following improvements may be noted:

- Both the HGS algorithms achieve more BKS than SISR (21 BKS) and KGLS (6) – see Table 9. Further, it should be noted that OR-Tools did not achieve any BKS.
- The HGS algorithms achieve much lower Avg. Gap followed by SISR (0.19%), KGLS (0.53%) and OR-Tools (4.01%).

Table 10 – Comparative results between OR-Tools, KGLS, SISR, HGS-CVRP, ICAHGS and BKS

Instances	OR-Tools (2020)			KGLS (2018)			SISR (2020)			HGS-CVRP (2020)			ICAHGS			BKS
	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	
X-n101-k25	27977.2	1.40	27865	27631.9	0.15	27595	27593.3	0.01	**27591**	**27591.0**	0.00	**27591**	**27591.0**	0.00	**27591**	27591
X-n106-k14	26757.5	1.50	26747	26413.2	0.19	26375	26380.9	0.07	26368	26381.4	0.07	26364	**26362.0**	0.00	**26362**	26362
X-n110-k13	15099.8	0.86	14986	**14971.0**	0.00	**14971**	14972.1	0.01	**14971**	**14971.0**	0.00	**14971**	**14971.0**	0.00	**14971**	14971
X-n115-k10	12808.3	0.48	12768	12747.1	0.00	**12747**	**12747.0**	0.00	**12747**	**12747.0**	0.00	**12747**	**12747.0**	0.00	**12747**	12747
X-n120-k6	13501.9	1.27	13458	**13332.0**	0.00	**13332**	**13332.0**	0.00	**13332**	**13332.0**	0.00	**13332**	**13332.0**	0.00	**13332**	13332
X-n125-k30	56853.4	2.37	56601	55740.8	0.36	55670	55559.8	0.04	**55539**	**55539.0**	0.00	**55539**	**55539.0**	0.00	**55539**	55539
X-n129-k18	29722.3	2.70	29668	28971.6	0.11	28954	28948.9	0.03	**28940**	**28940.0**	0.00	**28940**	**28940.0**	0.00	**28940**	28940
X-n134-k13	11171.0	2.34	11096	10940.5	0.22	10930	10937.7	0.20	10918	**10916.0**	0.00	**10916**	**10916.0**	0.00	**10916**	10916
X-n139-k10	13741.2	1.11	13693	**13590.0**	0.00	**13590**	13590.4	0.00	**13590**	**13590.0**	0.00	**13590**	**13590.0**	0.00	**13590**	13590
X-n143-k7	16135.6	2.77	16019	15730.6	0.19	15726	15727.8	0.18	**15700**	**15700.0**	0.00	**15700**	**15700.0**	0.00	**15700**	15700
X-n148-k46	44598.5	2.65	44334	43588.3	0.32	43507	43464.1	0.04	**43448**	**43448.0**	0.00	**43448**	**43448.0**	0.00	**43448**	43448
X-n153-k22	21789.3	2.68	21605	21386.0	0.78	21375	21228.6	0.04	21225	21225.0	0.02	21225	21225.0	0.02	**21220**	21220
X-n157-k13	17137.7	1.55	17086	16877.5	0.01	**16876**	16878.2	0.01	**16876**	**16876.0**	0.00	**16876**	**16876.0**	0.00	**16876**	16876
X-n162-k11	14262.2	0.88	14238	14147.0	0.06	14147	14159.0	0.15	**14138**	**14138.0**	0.00	**14138**	**14138.0**	0.00	**14138**	14138
X-n167-k10	21176.4	3.01	21158	20586.9	0.15	**20557**	20558.6	0.01	**20557**	**20557.0**	0.00	**20557**	**20557.0**	0.00	**20557**	20557
X-n172-k51	46874.9	2.78	46695	45802.8	0.43	45763	45622.6	0.03	**45607**	**45607.0**	0.00	**45607**	**45607.0**	0.00	**45607**	45607
X-n176-k26	49260.2	3.03	48986	47991.6	0.38	47958	47823.7	0.02	**47812**	**47812.0**	0.00	**47812**	**47812.0**	0.00	**47812**	47812
X-n181-k23	25935.6	1.43	25787	25602.3	0.13	25594	25575.1	0.02	**25569**	**25569.0**	0.00	**25569**	**25569.0**	0.00	**25569**	25569

Continuation of Table 10

Instances	OR-Tools (2020)			KGLS (2018)			SISR (2020)			HGS-CVRP (2020)			ICAHGS			BKS
	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	
X-n186-k15	24908.0	3.16	24908	24178.3	0.14	24156	24166.2	0.09	24151	**24145.0**	0.00	**24145**	**24145.0**	0.00	**24145**	24145
X-n190-k8	17421.9	2.60	17380	17033.5	0.32	17001	16982.8	0.02	**16980**	16983.3	0.02	**16980**	16981.7	0.01	**16980**	16980
X-n195-k51	46151.1	4.36	45757	44427.2	0.46	44396	44292.0	0.15	44241	**44225.0**	0.00	**44225**	**44225.0**	0.00	**44225**	44225
X-n200-k36	60447.9	3.19	60338	58828.0	0.43	58756	58635.6	0.10	**58587**	**58578.0**	0.00	**58578**	**58578.0**	0.00	**58578**	58578
X-n204-k19	20348.4	4.00	20212	19621.0	0.29	19581	19653.2	0.45	**19565**	**19565.0**	0.00	**19565**	**19565.0**	0.00	**19565**	19565
X-n209-k16	31775.5	3.65	31740	30709.7	0.18	30685	30661.7	0.02	**30656**	**30656.0**	0.00	**30656**	**30656.0**	0.00	**30656**	30656
X-n214-k11	11374.0	4.77	11228	10944.3	0.81	10913	10894.4	0.35	10874	10860.5	0.04	**10856**	10860.0	0.04	**10856**	10856
X-n219-k73	118038.0	0.38	117924	117689.1	0.08	117651	117623.7	0.02	117596	**117596.1**	0.00	**117595**	**117595.0**	0.00	**117595**	117595
X-n223-k34	42046.6	3.98	41794	40714.4	0.69	40686	40535.5	0.24	40504	**40437.0**	0.00	**40437**	**40437.0**	0.00	**40437**	40437
X-n228-k23	26613.4	3.39	26396	25836.8	0.37	25808	25814.3	0.28	25782	25742.8	0.00	**25742**	**25742.0**	0.00	**25742**	25742
X-n233-k16	19883.9	3.40	19682	19328.6	0.51	19268	19285.7	0.29	19232	**19230.0**	0.00	**19230**	**19230.0**	0.00	**19230**	19230
X-n237-k14	27927.5	3.27	27809	27095.9	0.20	27044	27081.1	0.14	27043	**27042.0**	0.00	**27042**	**27042.0**	0.00	**27042**	27042
X-n242-k48	85518.0	3.34	85518	83209.2	0.55	83136	82885.6	0.16	82805	82806.0	0.07	82771	82771.8	0.03	82764	82751
X-n247-k50	38282.8	2.71	37853	37388.4	0.31	37317	37379.6	0.28	**37274**	37277.1	0.01	**37274**	37275.0	0.00	**37274**	37274
X-n251-k28	40087.6	3.63	40007	38893.3	0.54	38847	38765.2	0.21	38687	38689.9	0.02	**38684**	**38684.0**	0.00	**38684**	38684
X-n256-k16	19294.5	2.42	19067	18891.6	0.28	18888	18887.3	0.26	18880	18839.6	0.00	**18839**	18839.4	0.00	**18839**	18839
X-n261-k13	27920.6	5.13	27760	26717.5	0.60	26671	26595.8	0.14	**26558**	26558.2	0.00	**26558**	26558.1	0.00	**26558**	26558
X-n266-k58	77660.8	2.89	77275	75954.6	0.63	75793	75609.2	0.17	75549	75564.7	0.11	**75478**	75545.9	0.09	75525	75478
X-n270-k35	36700.5	3.99	36401	35462.1	0.48	35447	35364.4	0.21	35325	35303.0	0.03	35303	35303.0	0.03	35303	35291

Continuation of Table 10

Instances	OR-Tools (2020)			KGLS (2018)			SISR (2020)			HGS-CVRP (2020)			ICAHGS			BKS
	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	
X-n275-k28	22087.3	3.96	21918	21299.4	0.26	21265	21250.5	0.03	**21245**	**21245.0**	0.00	**21245**	**21245.0**	0.00	**21245**	21245
X-n280-k17	35055.6	4.63	34859	33670.1	0.50	33598	33648.6	0.43	33545	33543.2	0.12	33506	33513.9	0.03	**33503**	33503
X-n284-k15	21137.9	4.57	20872	20360.0	0.72	20323	20287.6	0.36	20261	20245.5	0.15	20231	20236.0	0.10	**20215**	20215
X-n289-k60	98560.9	3.58	97868	95882.8	0.77	95770	95345.8	0.20	95245	95300.9	0.16	95242	95251.7	0.11	95204	95151
X-n294-k50	49301.8	4.54	49010	47454.1	0.62	47413	47251.9	0.19	47199	47184.1	0.05	47167	47167.6	0.01	**47161**	47161
X-n298-k31	36970.5	8.00	36296	34377.4	0.43	34359	34267.8	0.11	34234	34234.8	0.01	**34231**	**34231.0**	0.00	**34231**	34231
X-n303-k21	22573.7	3.85	22376	21903.4	0.77	21845	21772.9	0.17	21753	21748.5	0.06	21739	21741.3	0.02	21738	21736
X-n308-k13	27141.4	4.96	26934	26076.4	0.84	25999	26281.0	1.63	26224	25870.8	0.05	25862	25864.4	0.02	25861	25859
X-n313-k71	97497.4	3.67	96958	94763.8	0.77	94652	94155.7	0.12	94098	94112.2	0.07	94045	94060.9	0.02	94044	94043
X-n317-k53	79211.0	1.09	78863	78413.5	0.07	78391	78386.1	0.04	78361	78355.4	0.00	**78355**	**78355.0**	0.00	**78355**	78355
X-n322-k28	31488.5	5.55	30932	30038.0	0.68	30010	29892.5	0.20	29861	29848.7	0.05	**29834**	29848.0	0.05	**29834**	29834
X-n327-k20	28777.6	4.52	28592	27646.8	0.42	27613	27644.7	0.41	27611	27540.8	0.03	**27532**	27539.2	0.03	**27532**	27532
X-n331-k15	32648.2	4.97	32493	31200.1	0.32	31111	31124.5	0.07	31122	31103.0	0.00	**31102**	31103.0	0.00	**31102**	31102
X-n336-k84	143294.8	3.01	142905	140831.3	1.24	140716	139429.8	0.23	139272	139273.5	0.12	139205	139195.8	0.06	139139	139111
X-n344-k43	44036.4	4.72	43560	42350.5	0.71	42229	42122.7	0.17	42081	42075.6	0.06	42061	42058.4	0.02	42055	42050
X-n351-k40	27433.6	5.94	27093	26190.7	1.14	26150	25976.5	0.31	25965	25943.6	0.18	25924	25933.2	0.14	25924	25896
X-n359-k29	53858.4	4.57	53541	51901.3	0.77	51662	51549.8	0.09	51514	51620.0	0.22	51566	51554.1	0.10	51515	51505
X-n367-k17	23874.0	4.65	23597	22944.7	0.57	22867	22836.1	0.10	22821	**22814.0**	0.00	**22814**	**22814.0**	0.00	**22814**	22814
X-n376-k94	148775.7	0.72	148630	147854.1	0.10	147801	147763.5	0.03	147736	147714.5	0.00	**147713**	147715.5	0.00	**147713**	147713

Continuation of Table 10

Instances	OR-Tools (2020)			KGLS (2018)			SISR (2020)			HGS-CVRP (2020)			ICAHGS			BKS
	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	
X-n384-k52	69022.0	4.68	68550	66443.0	0.77	66363	66113.6	0.27	66046	66049.1	0.17	65997	65983.4	0.07	65968	65938
X-n393-k38	40785.6	6.60	40303	38466.4	0.54	38433	38384.5	0.33	38338	**38260.0**	0.00	**38260**	**38260.0**	0.00	**38260**	38260
X-n401-k29	68249.2	3.17	67913	66501.9	0.53	66466	66239.5	0.13	66222	66252.5	0.15	66209	66190.2	0.05	66166	66154
X-n411-k19	20810.6	5.57	20571	19924.8	1.08	19782	19776.7	0.33	19757	19720.3	0.04	19716	19718.7	0.03	**19712**	19712
X-n420-k130	111594.0	3.52	110857	108295.3	0.46	108175	107853.4	0.05	107809	107839.8	0.04	107810	107838.0	0.04	**107798**	107798
X-n429-k61	68858.4	5.21	68113	65857.5	0.62	65795	65539.3	0.14	65494	65502.7	0.08	65484	65457.3	0.01	**65449**	65449
X-n439-k37	37655.3	3.47	37171	36483.8	0.26	36445	36457.7	0.18	36402	36395.5	0.01	36395	36397.8	0.02	36395	36391
X-n449-k29	58427.1	5.78	58066	55770.7	0.97	55675	55388.8	0.28	55296	55368.5	0.25	55306	55322.0	0.16	55280	55233
X-n459-k26	25834.9	7.03	25435	24251.0	0.46	24234	24228.3	0.37	24187	24163.8	0.10	**24139**	24162.4	0.10	**24139**	24139
X-n469-k138	230963.3	4.12	230460	223468.0	0.74	223086	222253.9	0.19	222090	222170.1	0.16	221916	222073.6	0.11	221987	221824
X-n480-k70	92923.0	3.88	92457	89986.3	0.60	89926	89515.1	0.07	89458	89524.4	0.08	89498	89498.5	0.06	89464	89449
X-n491-k59	70817.2	6.52	69944	67145.6	1.00	67034	66606.9	0.19	66502	66641.5	0.24	66569	66558.3	0.11	66520	66483
X-n502-k39	70166.5	1.36	70032	69333.9	0.16	69307	69271.4	0.07	69238	69239.5	0.02	69230	69237.3	0.02	69228	69226
X-n513-k21	25845.9	6.80	25295	24360.7	0.66	24293	24293.9	0.38	24237	**24201.0**	0.00	**24201**	**24201.0**	0.00	**24201**	24201
X-n524-k153	156897.0	1.49	156322	155699.6	0.72	155422	154894.6	0.20	154758	154747.6	0.10	154646	154666.4	0.05	154610	154593
X-n536-k96	99575.6	4.99	98815	95864.7	1.07	95781	95145.9	0.32	95071	95091.9	0.26	95040	95052.0	0.22	95014	94846
X-n548-k50	89382.6	3.09	89066	86938.6	0.28	86901	86789.1	0.10	86710	86778.4	0.09	86710	86755.7	0.06	86727	86700
X-n561-k42	45758.6	7.12	45330	43031.7	0.74	42989	42875.0	0.37	42799	42742.7	0.06	42726	42742.3	0.06	42723	42717
X-n573-k30	52436.5	3.48	52080	50957.2	0.56	50849	50842.6	0.33	50777	50813.0	0.28	50757	50796.2	0.24	50741	50673

Continuation of Table 10

Instances	OR-Tools (2020)			KGLS (2018)			SISR (2020)			HGS-CVRP (2020)			ICAHGS			BKS
	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	
X-n586-k159	198347.2	4.22	197853	191411.4	0.58	191260	190640.1	0.17	190454	190588.1	0.14	190470	190484.1	0.09	190405	190316
X-n599-k92	113380.7	4.55	112831	109356.1	0.83	109125	108684.8	0.22	108598	108656.0	0.19	108605	108652.1	0.19	108586	108451
X-n613-k62	64073.6	7.62	63561	60201.2	1.12	60111	59705.6	0.29	59609	59696.3	0.27	59636	59660.3	0.21	59627	59535
X-n627-k43	64897.9	4.40	64590	62568.1	0.65	62486	62291.8	0.21	62221	62371.6	0.33	62238	62271.1	0.17	62242	62164
X-n641-k35	66862.3	4.99	66652	64094.3	0.64	63952	63851.8	0.26	63802	63874.2	0.30	63782	63830.3	0.23	63741	63684
X-n655-k131	107815.9	0.97	107710	106956.8	0.17	106936	106841.6	0.06	106808	106808.8	0.03	106785	106808.1	0.03	106802	106780
X-n670-k130	151874.1	3.79	151071	147654.2	0.90	147477	146961.5	0.43	146676	146777.7	0.30	146640	146732.1	0.27	146624	146332
X-n685-k75	74085.5	8.62	73090	68854.7	0.95	68628	68379.6	0.26	68271	68343.1	0.20	68288	68320.8	0.17	68273	68205
X-n701-k44	87060.3	6.27	86604	82513.5	0.72	82447	82053.9	0.16	82007	82237.3	0.38	82075	82121.0	0.24	82032	81923
X-n716-k35	46012.9	6.09	45704	43730.4	0.82	43627	43492.0	0.27	43449	43505.8	0.31	43459	43481.1	0.25	43451	43373
X-n733-k159	143829.1	5.61	142650	137299.3	0.82	137185	136445.2	0.19	136344	136426.9	0.18	136323	136389.9	0.15	136314	136187
X-n749-k98	82813.4	7.18	82083	78211.9	1.22	78109	77534.9	0.34	77399	77655.4	0.50	77563	77532.1	0.34	77350	77269
X-n766-k71	123106.2	7.59	121645	115186.0	0.67	115011	114836.0	0.37	114751	114764.5	0.30	114679	114719.0	0.26	114646	114417
X-n783-k48	77518.9	7.09	76764	73043.8	0.91	72974	72637.3	0.35	72544	72790.7	0.56	72704	72736.2	0.48	72619	72386
X-n801-k40	76428.2	4.26	76262	73590.5	0.39	73500	73412.0	0.15	73362	73500.4	0.27	73396	73450.1	0.20	73393	73305
X-n819-k171	165074.0	4.40	164377	159572.5	0.92	159396	158424.5	0.19	158344	158511.6	0.25	158391	158503.1	0.24	158368	158121
X-n837-k142	201836.8	4.18	201518	195135.0	0.72	194988	193946.6	0.11	193868	194231.3	0.26	194103	194252.2	0.27	194087	193737
X-n856-k95	91613.9	2.98	91109	89333.5	0.41	89218	89111.1	0.16	89042	89037.5	0.08	88986	89024.7	0.07	88990	88965
X-n876-k59	103576.1	4.31	103017	100115.7	0.82	100048	99484.5	0.19	99405	99682.7	0.39	99596	99675.2	0.38	99521	99299

Continuation of Table 10

Instances	OR-Tools (2020)			KGLS (2018)			SISR (2020)			HGS-CVRP (2020)			ICAHGS			BKS
	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	Avg	Gap	Best	
X-n895-k37	58191.7	8.04	57607	54306.0	0.83	54240	54072.3	0.39	53982	54070.6	0.39	54023	54070.5	0.39	53997	53860
X-n916-k207	342127.2	3.93	340947	331111.0	0.59	331006	329584.3	0.12	329418	329852.0	0.20	329572	329830.3	0.20	329671	329179
X-n936-k151	140479.3	5.85	139456	133831.4	0.84	133713	133497.1	0.59	133190	133369.9	0.49	133121	133085.7	0.28	133047	132715
X-n957-k87	88603.0	3.67	88222	85746.6	0.33	85656	85559.8	0.11	85493	85550.1	0.10	85506	85545.3	0.09	85513	85465
X-n979-k58	123885.2	4.13	123379	119600.1	0.52	119559	119108.2	0.11	119065	119247.5	0.23	119180	119244.2	0.23	119179	118976
X-n1001-k43	78084.7	7.92	77117	72998.9	0.89	72882	72533.1	0.25	72414	72748.8	0.54	72678	72651.5	0.41	72569	72355
Min Gap		0.38			**0.00**			**0.00**			**0.00**			**0.00**		
Avg Gap		4.01			0.54			0.20			0.11			**0.08**		
Max Gap		8.62			1.24			1.63			0.56			**0.48**		
Nb BKS			0			6			21			44			**51**	

4.4.7.1 Evolutionary performance of ICAHGS and HGS-CVRP

Figure 17 compares the Best performance of ICAHGS and HGS-CVRP over time - the same compiler and seed numbers are applied to each algorithm and Vidal's source code (VIDAL'S...,) is applied for HGS-CVRP. As shown, ICAHGS quickly shows superior performance to HGS-CVRP.

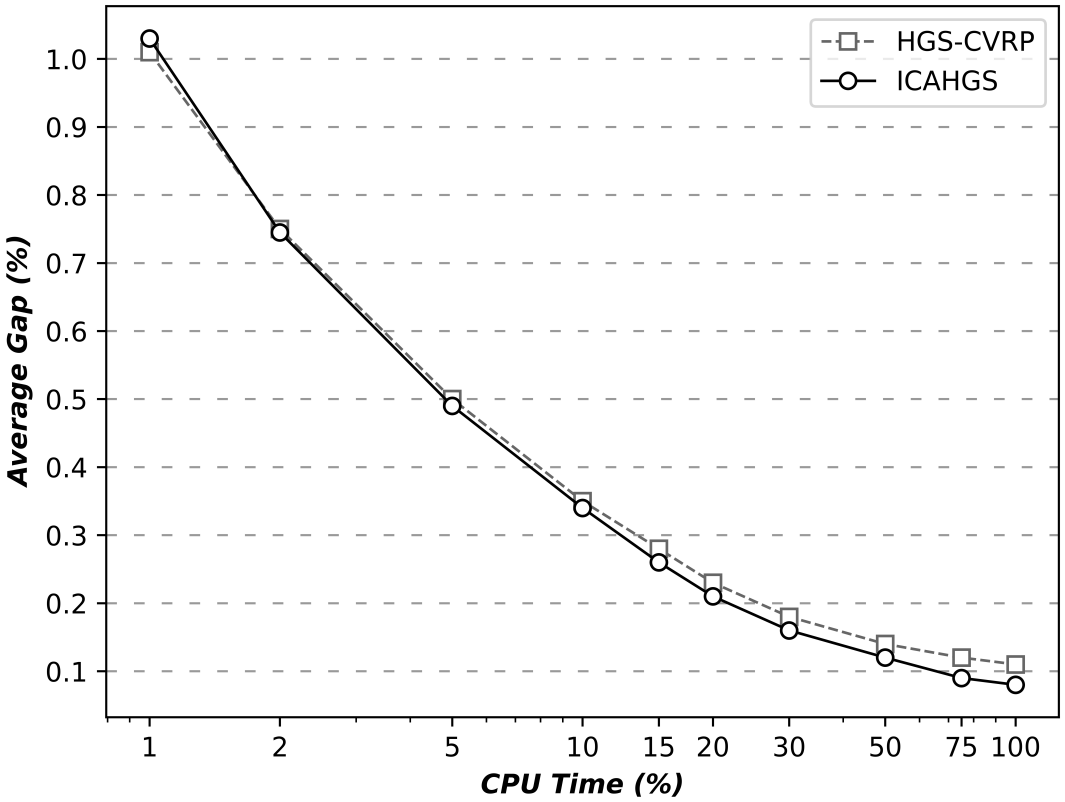

Figure 17 – Performance of ICAHGS and HGS-CVRP over time

Figures 18 to 21 highlight such performance over time for a) 2 cases of small instances and b) 2 cases of large instances. The route of each vehicle in the best-found solution by ICAHGS is displayed to the right of the evolutionary performance graphs. As shown in Figure 18, both algorithms obtained the BKS at the same time given the small size of the instance. However, as highlighted in Figure 19 and Figure 20, ICAHGS achieved the best solutions in approximately half the time consumed by the HGS-CVRP i.e. for both a small instance and also a large instance. However, in Figure 21, neither algorithm could find the BKS. HGS-CVRP has obtained the Gap 0.06 after 1484 seconds and then has stagnated. On the other hand, ICAHGS reached the Gap 0.06 in 332 seconds i.e., four times faster; and improvement continues until a superior Gap of 0.03 is achieved.

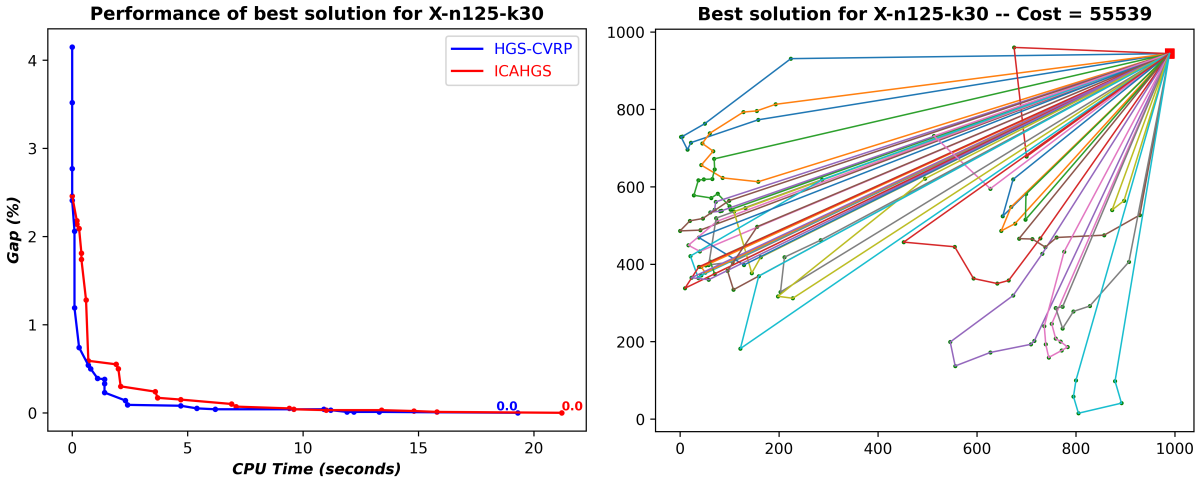

Figure 18 – Performance and best-found solution for X-n125-k30 (small)

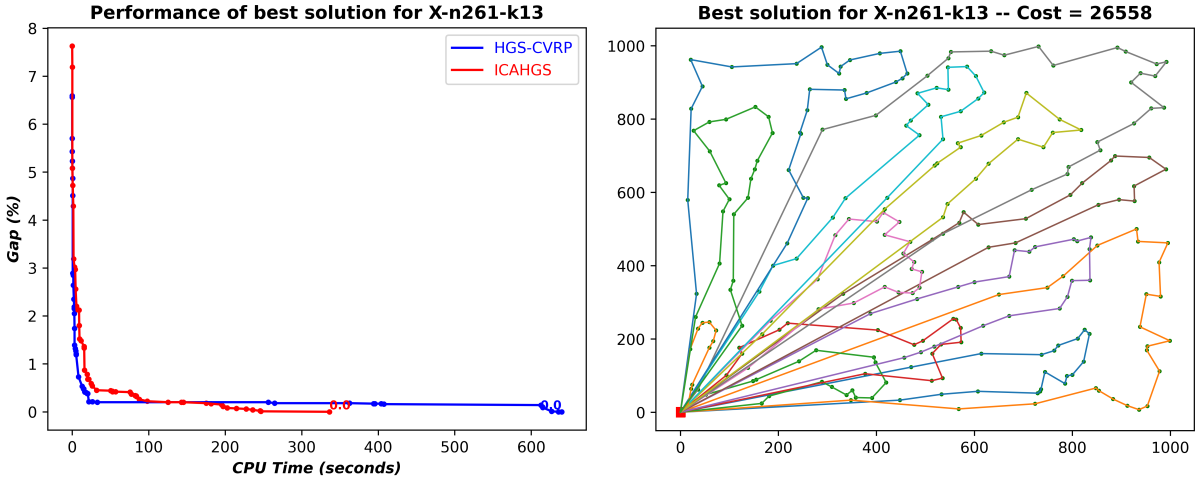

Figure 19 – Performance and best-found solution for X-n261-k13 (small)

4.4.8 Real-world application

Solving a real-world problem is the ultimate goal for VRP algorithms. Thus ICAHGS is compared to HGS-CVRP on 6 instances of the LoggiBUD benchmark (LOGGI. . . ,). The average and the best obtained results at T_{max} from 10 independent runs are provided in the Table 11. Table 12 provides a summary of the gaps in the average solution values for both the HGS-CVRP and ICAHGS algorithms as well as the number of BKS achieved. As shown, ICAHGS outperforms HGS-CVRP, achieving better performance in 5 out of the 6 instances for average solutions and 4 out of 6 instances for best solutions.

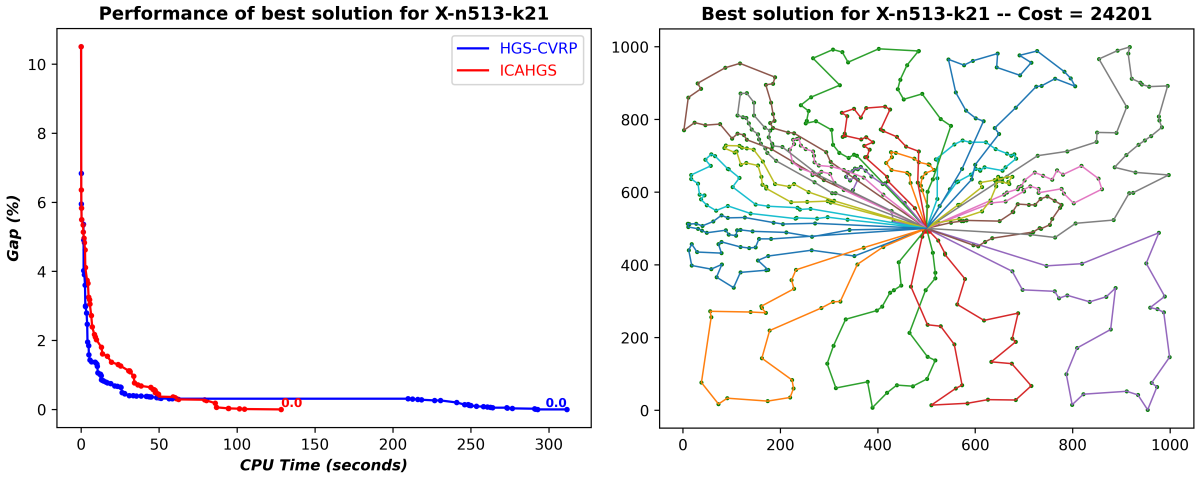

Figure 20 – Performance and best-found solution for X-n513-k21 (large)

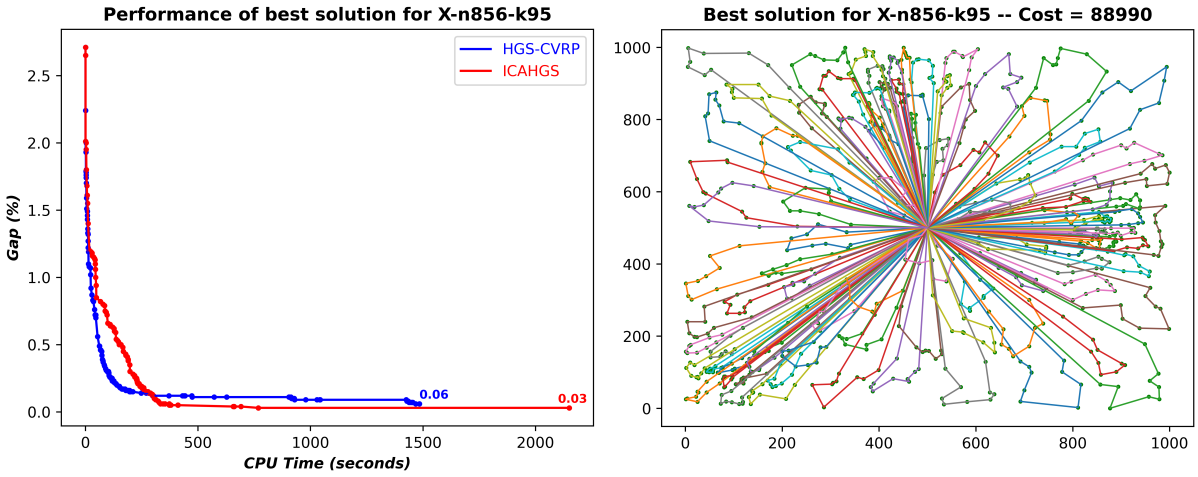

Figure 21 – Performance and best-found solution for X-n856-k95 (large)

Table 11 – Comparative results HGS-CVRP, ICAHGS and BKS for Loggi Benchmark for Urban Deliveries

Instance	HGS-CVRP (2020)				ICAHGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
Loggi-n401-k23	337231.8	0.08	337130	0.05	337153.2	0.06	337006	0.02	336946
Loggi-n501-k24	177506.2	0.02	177495	0.02	177505	0.02	177469	0.00	177466
Loggi-n601-k19	113309.4	0.14	113257	0.09	113270.2	0.10	113220	0.06	113155
Loggi-n601-k42	347092.8	0.01	**347059**	0.00	347081.2	0.01	**347059**	0.00	347059
Loggi-n901-k42	246614.8	0.08	246485	0.03	246661.8	0.10	246523	0.04	246418
Loggi-n1001-k31	286069.2	0.60	285743	0.49	285889.6	0.54	285649	0.45	284356

4.4.9 Statistical analysis

A t-test and the Wilcoxon test are both statistical tests that are used to compare the means of two groups of data. A t-test, also known as Student's t-test, is used to determine whether there is a significant difference between the means of two groups of

Table 12 – Summary of the results HGS-CVRP and ICAHGS on 6 instances of LoggiBUD benchmark

	HGS-CVRP	**ICAHGS**
Min Gap	0.01	0.01
Avg Gap	0.16	**0.14**
Max Gap	0.60	**0.54**
Number of BKS	1	1

data (STUDENT, 1908). On the other hand, the Wilcoxon test is a non-parametric test that is used when the data is not normally distributed or when the variances of the two groups are not equal (WILCOXON, 1947).

In order to evaluate the performance of the ICAHGS algorithm in comparison to state-of-the-art algorithms (OR-Tools, KGLS, SISR, and HGS-CVRP), both t-test and Wilcoxon test were conducted on the average of solutions obtained from the Uchoa benchmark (as presented in Table 10) - see Table 13. In the realm of statistical hypothesis testing, the null hypothesis is formally characterized as the assertion that "the population means of two groups are equivalent." In light of the computed p-value, which falls below the conventional significance threshold of 0.05, the null hypothesis is decisively rejected. This outcome signifies the existence of a statistically significant distinction between the two sets of data under scrutiny. Thus, as shown, with $\rho < 0.05$, the ICAHGS algorithm demonstrates significantly superior performance in comparison to the state-of-the-art algorithms.

Table 13 – Comparative statistical results between ICAHGS and state-of-the-art algorithms on Uchoa benchmark

	p-values			
	ICAHGS & OR-Tools	**ICAHGS & KGLS**	**ICAHGS & SISR**	**ICAHGS & HGS-CVRP**
t-test	<.001	<.001	<.001	<.001
Wilcoxon	<.001	<.001	<.001	<.001

Table 14 – Statistical results between ICAHGS and HGS-CVRP

	p-values	
Benchmarks	**t-test**	**Wilcoxon**
Golden	<.001	<.001
LoggiBUD	.015	.019
CMT	Nan	Nan

Table 14 presents the statistical analysis of ICAHGS performance compared to HGS-CVRP on the Golden, LoggiBUD and CMT benchmarks. Considering the same null hypothesis, ICAHGS shows a significant performance improvement over HGS-CVRP on the Golden and LoggiBUD benchmarks. However, for the CMT benchmark, both ICAHGS and HGS-CVRP were able to find almost all the BKS, resulting in no significant performance difference. It should be noted that, due to the lack of data behind the results for the state-of-the-art algorithms on these benchmarks, it was not possible to make any statistical comparison.

4.4.10 Complexity analysis

Time complexity analysis is a fundamental concept in computer science and is used to understand the performance of algorithms. It refers to the study of how the running time of an algorithm scales with the size of the input data. In this section, the time complexity of the proposed ICAHGS is compared to HGS-CVRP and the original ICA.

As outlined in Algorithm 5, the AssimRevol step of ICAHGS involves two loops for traversing empires and their colonies, during which OX Crossover, local search, and population management (if necessary) are applied to each colony. On the other hand, HGS-CVRP applies the same operations to each individual in the population, leading to the same time complexity as ICAHGS given the same initial population size. Additionally, ICAHGS includes the "imperialistic competition" process, which occurs frequently but does not contain any loops that would impact its time complexity calculations. It is worth noting that each iteration of ICAHGS is equivalent to $Pop \times RR$ iterations of HGS-CVRP, where Pop and RR refer to the initial population size and Revolution Rate, respectively.

ICAHGS with the original ICA and with the refined ICA, have the same time complexity. However, ICAHGS with the refined ICA exhibits a shorter run time. The run time for the original ICA can be calculated as $\mathcal{O}(Iter \times Emp \times Pop \times (1+RR))$, where $Iter$ is the maximum number of iterations, Emp represents the number of empires, Pop denotes the number of colonies, and RR is the Revolution Rate. On the other hand, the run time for the proposed ICAHGS with refined ICA is calculated as $\mathcal{O}(Iter \times Emp \times Pop \times RR)$. Given the same number of empires, colonies, and a Revolution Rate of 0.5 (as proposed in the ICAHGS), ICAHGS with refined ICA is 3 times faster. This improvement is achieved by combining the Assimilation and Revolution steps (see 4.1.2). In the original ICA, all colonies within each empire participate in the Assimilation step. In contrast, only $Pop \times RR$ colonies participate in the refined ICA.

5 DPIGA-HGS APPROACH

In Chapter 4, the performance evaluation demonstrated the superiority of the proposed ICAHGS (Imperialist Competitive Algorithm and Hybrid Genetic Search) over the original HGS and other state-of-the-art algorithms. Building upon this success, the focus of this chapter is to introduce a second approach called Dynamic Population Island GA and HGS (DPIGA-HGS). DPIGA-HGS is a novel hybrid metaheuristic model that combines a specialized island model (DPIGA) with HGS as its local search engine within each island, with the following objectives:

1. To introduce a novel variant of the island model, tailored to the specific requirements and characteristics of the problem domain, thereby contributing to the advancement of the field.

2. To further enhance the quality of the solutions in comparison to ICAHGS, thereby achieving even better optimization results.

5.1 Dynamic Population Island GA (DPIGA)

The proposed island model, drawing inspiration from the ICA, incorporates the fundamental characteristics of classical island models, but it distinguishes itself through its unique migration policy. As previously discussed in Section 2.8, the migration policy plays a crucial role in determining the selection criteria for migrants and the strategy for their replacement.

In the proposed DPIGA framework, during each migration interval, a single random individual (referred to as the *migrant*) is selected from the *Source-Island* and migrates to the *Target-Island*. The migrant is added to the population of the Target-Island, while being simultaneously removed from the population of the Source-Island. The Source-Island is identified as the island with the highest *IslandSolution*, which is calculated as the average fitness value of all feasible solutions within that particular island. On the other hand, the Target-Island is chosen randomly. This random selection of the target island contributes to maintaining diversification within the algorithm. Algorithm 6 presents the proposed migration policy.

To facilitate these migrations, a *Fully connected migration topology* is adopted, enabling each island to send migrants to all other islands and receive migrants from them. This fully interconnected topology facilitates efficient exchange of genetic information, allowing for exploration and exploitation across the entire island population.

Algorithm 6 proposed migration policy for DPIGA-HGS)

1: Select the island with the maximum *IslandSolution* as *SourceIsland*
2: Select one individual randomly from *SourceIsland*, as *migrant*
3: Select one island randomly as *TargetIsland*
4: Add the *migrant* to the population of the *TargetIsland*
5: Remove the *migrant* from the population of *SourceIsland*
6: **if** *SourceIsland* has no more individual in its population **then**
7: Remove *SourceIsland* from the model
8: **end if**

This migration scheme serves to gradually eliminate islands with the highest *IslandSolution* as individuals migrate between islands. Once an island loses all of its individuals, it is subsequently removed from the island model.

As the DPIGA is inspired by the ICA, specifically the refined version discussed in Section 4.1, there are both similarities and differences between the two approaches. In terms of similarities, the islands in DPIGA can be likened to the empires in the ICA, and the migration process in DPIGA corresponds to the concept of "Imperialistic competition" in the ICA. However, several differences distinguish DPIGA from ICA, as outlined below:

1. In the ICA, the number of colonies in an empire is determined by the power of its imperialist, whereas the initial population size of the islands in DPIGA is equal.

2. In the ICA, as discussed in Section 4.1, each empire must generate at least one feasible solution corresponding to its imperialist. However, the proposed DPIGA does not have such restrictions. The initial population in DPIGA can consist of various solution types, including feasible, infeasible, or a combination of both.

3. In the ICA, the cost of an empire is directly influenced by its imperialist, taking into account a small portion of the fitness value averages of its colonies (where $0 < \xi < 1$). In contrast, the proposed DPIGA incorporates the averages of fitness values of all feasible solutions. As a result, DPIGA maximizes the contribution of all feasible solutions.

5.2 Refined HGS-CVRP

As detailed in Section 4.2, the ICAHGS approach incorporates a multi-step restart mechanism to improve the quality of solutions. Experimental results demonstrate the beneficial impact of this modification on the obtained results. Leveraging the insights gained from these experiments, a new multi-step restart mechanism is developed and implemented in the DPIGA-HGS algorithm.

5.2.1 proposed multi-step restart mechanism

When designing a metaheuristic, it is crucial to consider two conflicting objectives: diversification and intensification (TALBI, 2009). The objective of the proposed multi-step restart mechanism is to strike a balance between these two criteria by enhancing intensification while preserving diversification. By incorporating this mechanism, the algorithm aims to intensify the search by focusing on promising areas of the solution space, while simultaneously ensuring diversity to explore different regions and avoid premature convergence.

Similar to the proposed multi-step restart mechanism in ICAHGS (see section 4.2.1), the newly designed mechanism includes two types: ***Partially*** and ***Totally***. However, the application levels of these types differ from ICAHGS. The proposed restart mechanism is outlined in Algorithm 7.

Algorithm 7 begins by creating and storing the *number of resets* and the *last reset value* (lines 1-8). The parameter *number of resets* determines whether *PartialRestart* will be performed (lines 10 and 12) or *TotalRestart* (line 14). *PartialRestart* consists of two steps: 25% and 10%. In the first step, 25% of the population is preserved, while the remaining 75% is regenerated randomly (line 10). This step is identical to the restart mechanism in ICAHGS. Additionally, another *PartialRestart* is applied when a second restart is required. In this step, 10% of the population is retained, and the remaining 90% is regenerated randomly (line 12). Figure 22 and Figure 23 depict the *PartiallyRestart* steps.

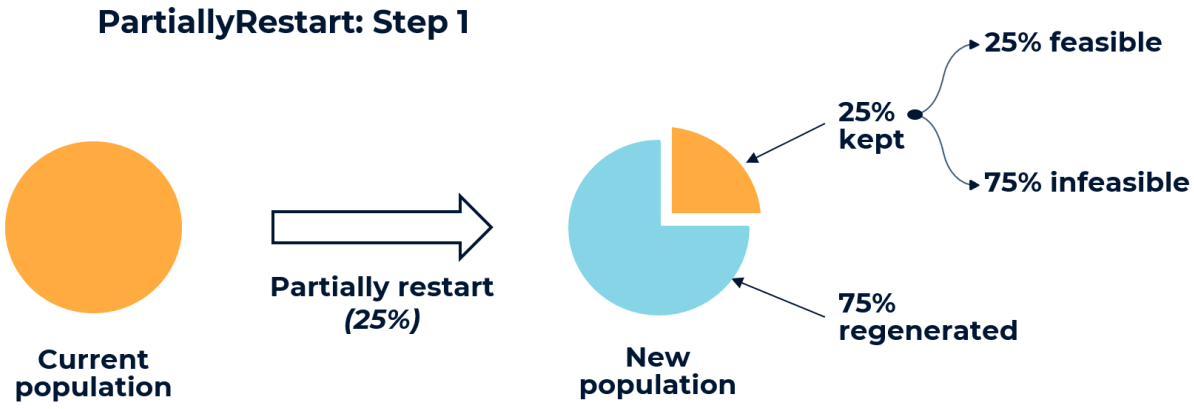

Figure 22 – The process of partially restart – Step 1

If no new solution is found after two *PartialRestarts*, a *TotalRestart* is triggered, where the entire population is regenerated randomly (line 9). *TotalRestart* aims to increase population diversity but may disrupt the balance between diversification and intensification. To restore the balance, lines 10-15 of the algorithm introduce different crossover operations when the number of restarts is even.

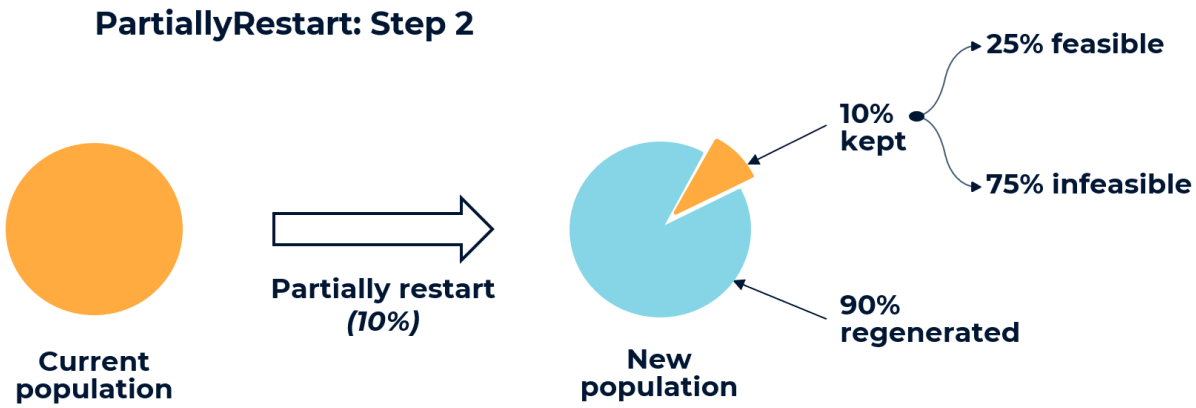

Figure 23 – The process of partially restart – Step 2

Algorithm 7 The proposed multi-step restart mechanism for DPIGA-HGS

Define variable *NumOfResets* to track the number of resets for each island
Define variable *LastResetValue* to keep the last *BestIslandSolution* cost when *Restart* mechanism is started
if $BestIslandSolution = LastResetValue$ **then**
  $NumOfResets = NumOfResets + 1$
else
  $LastResetValue = BestIslandSolution$
  $NumOfResets = 1$
end if
if $NumOfResets = 1$ **then**
  run *RestartPartially(0.25)*
else if $NumOfResets = 2$ **then**
  run *RestartPartially(0.10)*
else
  run *RestartTotally*
  if $NumOfResets \,\%\, 2 = 0$ **then**
    if $BestIslandSolution \neq BestSolutionOverall$ **then**
      $CrossType = True$
    else
      $CrossType = False$
    end if
  else
    $CrossType = False$
  end if
end if

5.3 DPIGA-HGS approach

An overview of the proposed DPIGA-HGS algorithm is provided in Figure 24. The algorithm employs an island model as the main evolutionary algorithm, which is based on the proposed DPIGA method discussed in section 5.1. Within each island of the DPIGA model, the individuals are managed by a refined HGS-CVRP, as described in section 5.2.

The DPIGA-HGS algorithm is presented in Algorithm 8. In this algorithm, the refined HGS-CVRP is applied to the individuals in lines 11-15, while the proposed migration policy is implemented in line 18. During each iteration of the algorithm and for each island, the number of mutated individuals (*NumberOfMutatedIndv*) is calculated based on the *mutation rate* (line 5), and the crossover operator is applied in lines 6-10. Crossover is performed either between randomly selected colonies or between the *BestSolutionOverall* and randomly selected colonies, controlled by the parameter *CrossType*.

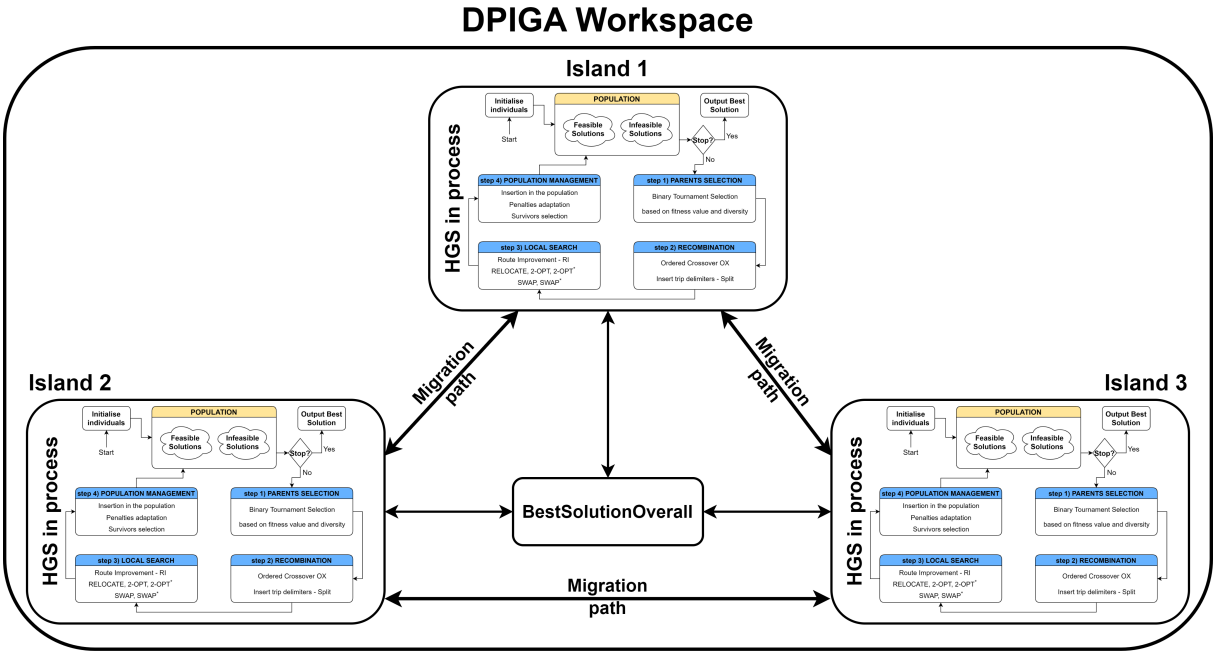

Figure 24 – An overview of the proposed DPIGA-HGS

5.4 Experimental results

In this section, the experimental results are presented and analyzed to compare the performance of the two proposed algorithms: ICAHGS and DPIGA-HGS. To ensure fair and consistent evaluation, the same benchmark instances are utilized for both algorithms. However, it should be noted that while ICAHGS has been previously compared with HGS-CVRP and other state-of-the-art algorithms, this section focuses solely on comparing DPIGA-HGS with ICAHGS. It is important to highlight that the software and hardware configurations remain unchanged throughout the experimentation process, ensuring that both algorithms are assessed under identical environments.

5.4.1 Tuning the proposed IGA parameters

Two distinct subsets of parameters, namely "Initial Islands" ($subset_1 = [3, 4, 5]$) and "Mutation Rate" ($subset_2 = [0.4, 0.5, 0.6]$), were defined for experimental purposes.

Algorithm 8 DPIGA-HGS

```
Start
Create initial population and islands
while stopping criterion is not met do
  for each island in islands do
    for ii = 1 to NumberOfMutatedIndv do
      if CrossType = False then
        do CrossOver between individuals
      else
        do CrossOver between BestSolutionOverall and a random selected individual
      end if
      Local Search
      Population management & penalties adaptation
      if no improvement after nbIter iterations then
        run Restart Mechanism
      end if
    end for
  end for
  After a preset migration interval, perform Migration process
end while
Output the best-found solution
End
```

The DPIGA-HGS algorithm was executed for 10 independent runs, each with a different seed number, utilizing the same problem instances as described in section 4.4.3. The HGS-CVRP parameters were kept unchanged from (VIDAL, 2022). The average performance, indicated by the Value, and the average efficiency (Time), measured in terms of consumed CPU time, were calculated and documented in Table 15. The optimal values obtained for the "Initial Islands" (II) and "Mutation Rate" (MR) parameters were found to be 4 and 0.5, respectively. The overview of the DPIGA-HGS parameters is provided in Table 16.

Table 15 – Sweep of DPIGA Parameter

Cases		**X-n129-k18**		**X-n228-k23**		**X-n336-k84**		**X-n524-k153**		**X-n801-k40**	
II	**RR**	**Value**	**time**	**Value**	**time**	**Value**	**time**	**Value**	**time**	**Value**	**time**
3	0.4	28940.0	9.70	25743.0	70.90	139238.6	721.52	154737.1	860.78	73547.0	1965.59
3	0.5	28940.0	12.60	25743.0	49.92	139320.9	857.06	154680.4	909.87	73586.0	1970.08
3	0.6	28940.0	9.20	25742.0	277.80	139338.6	806.55	154694.1	1192.52	73452.5	2010.80
4	0.4	28940.0	17.42	25742.0	261.59	139254.2	711.24	154731.4	861.23	73457.0	2025.23
4	0.5	28940.0	7.98	25742.0	275.24	139191.3	705.14	154624.6	935.36	73450.1	2032.96
4	0.6	28940.0	24.86	25742.0	180.50	139223.4	808.84	154675.3	1217.40	73453.0	1850.58
5	0.4	28940.0	15.63	25743.0	64.63	139378.9	676.27	154749.6	1207.36	73512.5	1876.47
5	0.5	28940.0	32.34	25743.0	30.52	139274.4	911.61	154674.4	1029.19	73459.6	1841.05
5	0.6	28940.0	10.03	25742.0	154.47	139335.7	737.38	154684.5	1247.46	73526.3	1803.83

Table 16 – The parameters of DPIGA-HGS

	Parameter name	Additional comment	Value
DPIGA	Initial population	From the original HGS-CVRP	100
	Initial islands	From parameter tuning	4
	Mutation Rate	From parameter tuning	0.5
HGS-CVRP	μ (miu)	Minimum population size	25
	λ (lambda)	Generation size	40
	n_{elite}	Number of elite solutions considered in the fitness calculation	4
	$n_{closest}$	Number of close solutions considered in the diversity-contribution measure	5
	Γ	Granular search parameter	20
	ξ^{Ref}	Target proportion of feasible individuals for penalty adaptation	0.2

5.4.2 DPIGA vs. original IGA

In order to comprehensively evaluate the performance and efficiency of the proposed DPIGA-HGS, a comparative analysis involving two distinct approaches is employed: the IGAHGS with the original IGA (detailed in section 2.8) and refined HGS (described in section 5.2), and the DPIGA-HGS itself, described in section 5.1.

The main difference between DPIGA and the original IGA lies in the employed migration policy within DPIGA. To facilitate the assessment, the following migration policy and parameters are adopted for the original IGA:

- One random individual will be migrated from each island to other islands.
- At the end of each migration interval and to maintain the equality of populations, $NumberOfIslands - 1$ worst individuals (based on fitness value) will be removed from the population of each island.

This comparative analysis is conducted on 10 randomly selected benchmark instances (originating from section 4.4.4), and the resulting outcomes are documented in Table 17. In the table, if BKS is reached, BKS is listed as the improvement; otherwise, the % improvement is listed either with respect to the value achieved or to the gap. If both versions achieved BKS i.e., no improvement is needed, the effect on time is given.

Through the implementation of DPIGA-HGS, a noteworthy accomplishment of achieving 4 BKS instances is observed. Notably, in 5 instances, both the original IGA and DPIGA-HGS achieved the BKS; however, the latter demonstrated better efficiency. Additionally, for the X-n856-k95 instance, the performance of DPIGA-HGS, concerning the gap metric, exhibited an improvement of over 56%.

Table 17 – Comparative results to analyze the refined IGA

Instances	Original IGA with refined HGS			DPIGA-HGS			improvement	
	Best	Gap	time	Best	Gap	time	type	percent
X-n106-k14	26362	0.00	213.26	26362	0.00	161.74	Time	24.16
X-n125-k30	55539	0.00	23.09	55539	0.00	16.04	Time	30.53
X-n261-k13	26558	0.00	614.83	26558	0.00	146.50	Time	76.17
X-n284-k15	20247	0.16	562.51	20215	0.00	701.41	BKS	-
X-n317-k53	78355	0.00	372.89	78355	0.00	165.95	Time	55.50
X-n411-k19	19718	0.03	1148.24	19712	0.00	437.53	BKS	-
X-n429-k61	65480	0.05	645.063	65449	0.00	1046.72	BKS	-
X-n459-k26	24153	0.06	711.676	24139	0.00	502.31	BKS	-
X-n513-k21	24201	0.00	180.649	24201	0.00	143.76	Time	20.42
X-n856-k95	89006	0.05	2147.907	88983	0.02	586.84	Gap	56.10

5.4.3 Analysis of multi-step restart mechanism (MSRM)

To investigate the impact of the proposed multi-step restart mechanism (MSRM), as detailed in section 5.2.1, a comparative analysis of the performance and efficiency of the DPIGA-HGS algorithm with and without the MSRM is conducted. The former allows for both partial and total restarts, while the latter permits only total restarts. The results of this assessment were obtained using the same set of 10 randomly selected instances, as in section 5.4.2, and are presented in Table 18.

Notably, the integration of the MSRM led to significant improvements in 5 instances, resulting in the attainment of BKS. Moreover, in 4 instances where both approaches achieved BKS, the application of MSRM further enhanced efficiency. However, in the case of X-n856-k95, no additional improvement was observed, as both algorithms had already reached the same solutions before incorporating the MSRM.

5.4.4 DPIGA-HGS versus ICAHGS on CMT and Golden benchmarks

To study the generality of the results, two further benchmarks were examined: CMT (CHRISTOFIDES, 1979) and Golden (GOLDEN et al., 1998). Table 20 (CMT) and Table 21 (Golden) provide the average and best results obtained at T_{max} from 10 independent runs. In Table 19, a summary of the gaps in the average values of solutions among all instances of CMT and Golden benchmarks is presented, along with the number of BKS instances discovered.

A total of 13 BKS instances out of 14 were found by both DPIGA-HGS and ICAHGS for the CMT dataset, with the gap for the remaining instance (CMT5) being

Table 18 – Comparative results to analyze the multi-step restart mechanism

Instances	DPIGA-HGS (without MSRM)			DPIGA-HGS (with MSRM)			improvement	
	Best	Gap	time	Best	Gap	time	type	percent
X-n106-k14	26378	0.06	221.58	26362	0.00	161.74	BKS	-
X-n125-k30	55539	0.00	16.38	55539	0.00	16.04	Time	2.08
X-n261-k13	26558	0.00	323.52	26558	0.00	146.50	Time	54.72
X-n284-k15	20228	0.06	848.82	20215	0.00	701.41	BKS	-
X-n317-k53	78355	0.00	208.52	78355	0.00	165.95	Time	20.41
X-n411-k19	19718	0.03	1008.44	19712	0.00	437.53	BKS	-
X-n429-k61	65468	0.03	1260.94	65449	0.00	1046.72	BKS	-
X-n459-k26	24163	0.10	1367.97	24139	0.00	502.31	BKS	-
X-n513-k21	24201	0.00	158.24	24201	0.00	143.76	Time	9.15
X-n856-k95	88983	0.02	586.84	88983	0.02	586.84	-	-

Table 19 – Comparative results between DPIGA-HGS and ICAHGS on CMT and Golden benchmarks

	CMT		Golden	
	ICAHGS	DPIGA-HGS	ICAHGS	DPIGA-HGS
Min Gap	0.00	0.00	0.00	0.00
Avg Gap	0.0009	0.0009	0.14	0.13
Max Gap	0.0123	0.0123	0.56	0.55
Number of BKS	13	13	8	8

0.01% - negligible for most applications.

Regarding the Golden dataset, 8 BKS were achieved by both algorithms. DPIGA-HGS performed better for 5 and 1 instances in terms of Average and Best found solutions, respectively (green cells in Table 21), while ICAHGS showed better Average for 2 instances (red cells in Table 21). The differences between the obtained results were too small and negligible for most applications, emphasizing the comparable performance of both algorithms.

5.4.5 DPIGA-HGS versus ICAHGS on Uchoa benchmarks

DPIGA-HGS experiments were conducted on 100 benchmark instances from the Uchoa benchmark (UCHOA et al., 2017) and obtained results were compared with ICAHGS results detailed in section 4.4.7. A detailed comparison of the gaps in average solution values for both ICAHGS and DPIGA-HGS is presented in Table 22. For a comprehensive assessment of the performance of DPIGA-HGS against other state-of-the-art algorithms, as detailed in Table 23, Table 23 provides a summary of the gaps in average solution values

Table 20 – Comparative results ICAHGS, DPIGA-HGS and BKS for CMT benchmark

Instance	ICAHGS				DPIGA-HGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
CMT1	**524.61**	0.00	**524.61**	0.00	**524.61**	0.00	**524.61**	0.00	524.61
CMT2	**835.26**	0.00	**835.26**	0.00	**835.26**	0.00	**835.26**	0.00	835.26
CMT3	**826.14**	0.00	**826.14**	0.00	**826.14**	0.00	**826.14**	0.00	826.14
CMT4	**1028.42**	0.00	**1028.42**	0.00	**1028.42**	0.00	**1028.42**	0.00	1028.42
CMT5	1291.45	0.01	1291.45	0.01	1291.45	0.01	1291.45	0.01	1291.29
CMT6	**555.43**	0.00	**555.43**	0.00	**555.43**	0.00	**555.43**	0.00	555.43
CMT7	**909.68**	0.00	**909.68**	0.00	**909.68**	0.00	**909.68**	0.00	909.68
CMT8	**865.95**	0.00	**865.95**	0.00	**865.95**	0.00	**865.95**	0.00	865.95
CMT9	**1162.55**	0.00	**1162.55**	0.00	**1162.55**	0.00	**1162.55**	0.00	1162.55
CMT10	**1395.85**	0.00	**1395.85**	0.00	**1395.85**	0.00	**1395.85**	0.00	1395.85
CMT11	**1042.12**	0.00	**1042.12**	0.00	**1042.12**	0.00	**1042.12**	0.00	1042.12
CMT12	**819.56**	0.00	**819.56**	0.00	**819.56**	0.00	**819.56**	0.00	819.56
CMT13	**1541.14**	0.00	**1541.14**	0.00	**1541.14**	0.00	**1541.14**	0.00	1541.14
CMT14	**866.37**	0.00	**866.37**	0.00	**866.37**	0.00	**866.37**	0.00	866.37

Table 21 – Comparative results ICAHGS, DPIGA-HGS and BKS for Golden benchmark

Instance	ICAHGS				DPIGA-HGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
Golden_1	5623.47	0.00	5623.47	0.00	5623.47	0.00	5623.47	0.00	5623.47
Golden_2	8417.20	0.15	8406.33	0.02	8417.20	0.15	8406.33	0.02	8404.61
Golden_3	11036.20	0.35	11036.20	0.35	11036.20	0.35	11036.20	0.35	10997.80
Golden_4	13624.50	0.26	13624.50	0.26	13624.50	0.26	13624.50	0.26	13588.60
Golden_5	6460.98	0.00	6460.98	0.00	6460.98	0.00	6460.98	0.00	6460.98
Golden_6	8412.90	0.15	8412.90	0.15	8412.03	0.14	8404.20	0.05	8400.33
Golden_7	10159.55	0.56	10132.90	0.30	10158.07	0.55	10132.90	0.30	10102.70
Golden_8	11635.30	0.00	11635.30	0.00	11635.30	0.00	11635.30	0.00	11635.30
Golden_9	580.30	0.10	579.71	0.00	580.49	0.14	579.71	0.00	579.70
Golden_10	737.39	0.27	736.00	0.08	737.15	0.23	736.00	0.08	735.43
Golden_11	913.60	0.18	913.06	0.12	913.60	0.18	913.06	0.12	911.98
Golden_12	1104.74	0.32	1103.34	0.19	1104.10	0.26	1103.34	0.19	1101.24
Golden_13	857.19	0.00	857.19	0.00	857.19	0.00	857.19	0.00	857.19
Golden_14	1080.55	0.00	1080.55	0.00	1080.55	0.00	1080.55	0.00	1080.55
Golden_15	1339.36	0.16	1337.84	0.04	1339.36	0.16	1337.84	0.04	1337.27
Golden_16	1614.50	0.20	1612.57	0.08	1614.49	0.20	1612.57	0.08	1611.28
Golden_17	707.76	0.00	707.76	0.00	707.76	0.00	707.76	0.00	707.76
Golden_18	995.13	0.00	995.13	0.00	995.13	0.00	995.13	0.00	995.13
Golden_19	1365.65	0.00	1365.60	0.00	1365.65	0.00	1365.60	0.00	1365.60
Golden_20	1818.37	0.04	1817.89	0.02	1818.44	0.05	1817.89	0.02	1817.59

across all instances and the number of BKS instances found. Furthermore, a more in-depth analysis of the comparative results between ICAHGS and DPIGA-HGS is presented in

Table 24.

The superiority of DPIGA-HGS in terms of solution quality compared to ICAHGS and other state-of-the-art algorithms is evident. Several notable improvements over ICAHGS are worth mentioning:

- DPIGA-HGS achieved a remarkable 8.9% advantage over ICAHGS concerning the number of found, reaching an impressive total of 56 BKS compared to 51 BKS for ICAHGS (see Table 23).
- DPIGA-HGS demonstrated significant improvements of 16.1% and 12.5% in the Max. and Avg. Gap of Average solutions, respectively (see Table 22).
- Considering small instances (the first 50 instances), DPIGA-HGS yielded optimal or near-optimal solutions, with 2.31% improvement in the Max. Gap and a noteworthy 19.81% improvement in the Avg. Gap of Average solutions relative to ICAHGS (see Table 22).
- For large instances (the last 50 instances), DPIGA-HGS excelled with a 16.1% improvement in the Max. Gap and an 11.94% improvement in the Avg. Gap of average solutions when compared to ICAHGS (see Table 22).

Table 22 – Comparing Gaps of average solution values for ICAHGS and DPIGA-HGS

Gaps	**included instances**	**ICAHGS**	**DPIGA-HGS**	**improvement (%)**
Max Gap	100	0.48	0.41	16.1
Avg. Gap	100	0.08	0.07	12.5
Max Gap	First 50	0.11	0.10	2.31
Avg. Gap	First 50	0.012	0.009	19.81
Max Gap	Last 50	0.48	0.41	16.10
Avg. Gap	Last 50	0.151	0.133	11.94

Table 23 – Comparative results between different algorithms

	OR-Tools (2020)	**KGLS (2018)**	**SISR (2020)**	**HGS-CVRP (2020)**	**ICAHGS**	**DPIGA-HGS**
Min Gap	0.38	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Avg Gap	4.01	0.53	0.19	0.11	0.08	**0.07**
Max Gap	8.62	1.24	1.63	0.56	0.48	**0.41**
Number of BKS	0	6	21	44	51	**56**

Table 24 – Comparative results between ICAHGS, DPIGA-HGS and BKS

Instance	ICAHGS				DPIGA-HGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
X-n101-k25	**27591.0**	0.00	**27591**	0.00	**27591.0**	0.00	**27591**	0.00	27591
X-n106-k14	**26362.0**	0.00	**26362**	0.00	**26362.0**	0.00	**26362**	0.00	26362
X-n110-k13	**14971.0**	0.00	**14971**	0.00	**14971.0**	0.00	**14971**	0.00	14971
X-n115-k10	**12747.0**	0.00	**12747**	0.00	**12747.0**	0.00	**12747**	0.00	12747
X-n120-k6	**13332.0**	0.00	**13332**	0.00	**13332.0**	0.00	**13332**	0.00	13332
X-n125-k30	**55539.0**	0.00	**55539**	0.00	**55539.0**	0.00	**55539**	0.00	55539
X-n129-k18	**28940.0**	0.00	**28940**	0.00	**28940.0**	0.00	**28940**	0.00	28940
X-n134-k13	**10916.0**	0.00	**10916**	0.00	**10916.0**	0.00	**10916**	0.00	10916
X-n139-k10	**13590.0**	0.00	**13590**	0.00	**13590.0**	0.00	**13590**	0.00	13590
X-n143-k7	**15700.0**	0.00	**15700**	0.00	**15700.0**	0.00	**15700**	0.00	15700
X-n148-k46	**43448.0**	0.00	**43448**	0.00	**43448.0**	0.00	**43448**	0.00	43448
X-n153-k22	21224.5	0.02	**21220**	0.00	21224.5	0.02	**21220**	0.00	21220
X-n157-k13	**16876.0**	0.00	**16876**	0.00	**16876.0**	0.00	**16876**	0.00	16876
X-n162-k11	**14138.0**	0.00	**14138**	0.00	**14138.0**	0.00	**14138**	0.00	14138
X-n167-k10	**20557.0**	0.00	**20557**	0.00	**20557.0**	0.00	**20557**	0.00	20557
X-n172-k51	**45607.0**	0.00	**45607**	0.00	**45607.0**	0.00	**45607**	0.00	45607
X-n176-k26	**47812.0**	0.00	**47812**	0.00	**47812.0**	0.00	**47812**	0.00	47812
X-n181-k23	**25569.0**	0.00	**25569**	0.00	**25569.0**	0.00	**25569**	0.00	25569
X-n186-k15	**24145.0**	0.00	**24145**	0.00	**24145.0**	0.00	**24145**	0.00	24145
X-n190-k8	16981.7	0.01	**16980**	0.00	16981.6	0.01	**16980**	0.00	16980
X-n195-k51	**44225.0**	0.00	**44225**	0.00	**44225.0**	0.00	**44225**	0.00	44225
X-n200-k36	**58578.0**	0.00	**58578**	0.00	**58578.0**	0.00	**58578**	0.00	58578
X-n204-k19	**19565.0**	0.00	**19565**	0.00	**19565.0**	0.00	**19565**	0.00	19565
X-n209-k16	**30656.0**	0.00	**30656**	0.00	**30656.0**	0.00	**30656**	0.00	30656
X-n214-k11	10860.0	0.04	**10856**	0.00	**10856.0**	0.00	**10856**	0.00	10856
X-n219-k73	**117595.0**	0.00	**117595**	0.00	**117595.0**	0.00	**117595**	0.00	117595
X-n223-k34	**40437.0**	0.00	**40437**	0.00	**40437.0**	0.00	**40437**	0.00	40437
X-n228-k23	**25742.0**	0.00	**25742**	0.00	**25742.0**	0.00	**25742**	0.00	25742
X-n233-k16	**19230.0**	0.00	**19230**	0.00	**19230.0**	0.00	**19230**	0.00	19230

Continuation of Table 24

Instance	ICAHGS				DPIGA-HGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
X-n237-k14	**27042.0**	0.00	**27042**	0.00	**27042.0**	0.00	**27042**	0.00	27042
X-n242-k48	82771.8	0.03	82764	0.02	82771.3	0.02	**82751**	0.00	82751
X-n247-k50	37275.0	0.00	**37274**	0.00	**37274.0**	0.00	**37274**	0.00	37274
X-n251-k28	**38684.0**	0.00	**38684**	0.00	**38684.0**	0.00	**38684**	0.00	38684
X-n256-k16	18839.4	0.00	**18839**	0.00	**18839.0**	0.00	**18839**	0.00	18839
X-n261-k13	26558.1	0.00	**26558**	0.00	**26558.0**	0.00	**26558**	0.00	26558
X-n266-k58	75545.9	0.09	75525	0.06	75545.1	0.09	**75478**	0.00	75478
X-n270-k35	35303.0	0.03	35303	0.03	35302.5	0.03	35298	0.02	35291
X-n275-k28	**21245.0**	0.00	**21245**	0.00	**21245.0**	0.00	**21245**	0.00	21245
X-n280-k17	33513.9	0.03	**33503**	0.00	33512.2	0.03	**33503**	0.00	33503
X-n284-k15	20236.0	0.10	**20215**	0.00	20235.9	0.10	**20215**	0.00	20215
X-n289-k60	95251.7	0.11	95204	0.06	95245.7	0.10	95194	0.05	95151
X-n294-k50	47167.6	0.01	**47161**	0.00	47167.6	0.01	**47161**	0.00	47161
X-n298-k31	**34231.0**	0.00	**34231**	0.00	**34231.0**	0.00	**34231**	0.00	34231
X-n303-k21	21741.3	0.02	21738	0.01	21740.2	0.02	21738	0.01	21736
X-n308-k13	25864.4	0.02	25861	0.01	25862.1	0.01	**25859**	0.00	25859
X-n313-k71	94060.9	0.02	94044	0.00	94060.9	0.02	94044	0.00	94043
X-n317-k53	**78355.0**	0.00	**78355**	0.00	**78355.0**	0.00	**78355**	0.00	78355
X-n322-k28	29848.0	0.05	**29834**	0.00	29839.6	0.02	**29834**	0.00	29834
X-n327-k20	27539.2	0.03	**27532**	0.00	27532.9	0.00	**27532**	0.00	27532
X-n331-k15	31103.0	0.00	**31102**	0.00	31102.9	0.00	**31102**	0.00	31102
X-n336-k84	139195.8	0.06	139139	0.02	139191.3	0.06	139139	0.02	139111
X-n344-k43	42058.4	0.02	42055	0.01	42058.4	0.02	42055	0.01	42050
X-n351-k40	25933.2	0.14	25924	0.11	25933.2	0.14	25919	0.09	25896
X-n359-k29	51554.1	0.10	51515	0.02	51554.1	0.10	51515	0.02	51505
X-n367-k17	**22814.0**	0.00	**22814**	0.00	**22814.0**	0.00	**22814**	0.00	22814
X-n376-k94	147715.5	0.00	**147713**	0.00	147714.5	0.00	**147713**	0.00	147713
X-n384-k52	65983.4	0.07	65968	0.05	65983.4	0.07	65968	0.05	65938
X-n393-k38	**38260.0**	0.00	**38260**	0.00	**38260.0**	0.00	**38260**	0.00	38260

Continuation of Table 24

Instance	ICAHGS				DPIGA-HGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
X-n401-k29	66190.2	0.05	66166	0.02	66190.2	0.05	66166	0.02	66154
X-n411-k19	19718.7	0.03	**19712**	0.00	19717.5	0.03	**19712**	0.00	19712
X-n420-k130	107838.0	0.04	**107798**	0.00	107822.5	0.02	**107798**	0.00	107798
X-n429-k61	65457.3	0.01	**65449**	0.00	65457.3	0.01	**65449**	0.00	65449
X-n439-k37	36395.5	0.01	36395	0.01	36394.6	0.01	**36391**	0.00	36391
X-n449-k29	55322.0	0.16	55280	0.09	55322.0	0.16	55280	0.09	55233
X-n459-k26	24162.4	0.10	**24139**	0.00	24147.2	0.03	**24139**	0.00	24139
X-n469-k138	222073.6	0.11	221987	0.07	222066.0	0.11	221897	0.03	221824
X-n480-k70	89498.5	0.06	89464	0.02	89466.0	0.02	89458	0.01	89449
X-n491-k59	66558.3	0.11	66520	0.06	66558.3	0.11	66556	0.11	66483
X-n502-k39	69237.3	0.02	69228	0.00	69233.6	0.01	69227	0.00	69226
X-n513-k21	**24201.0**	0.00	**24201**	0.00	**24201.0**	0.00	**24201**	0.00	24201
X-n524-k153	154666.4	0.05	154610	0.01	154624.6	0.02	154605	0.01	154593
X-n536-k96	95052.0	0.22	95014	0.18	95045.8	0.21	94988	0.15	94846
X-n548-k50	86741.9	0.05	86707	0.01	86729.3	0.03	86704	0.00	86700
X-n561-k42	42742.3	0.06	42723	0.01	42725.8	0.02	42719	0.00	42717
X-n573-k30	50796.2	0.24	50741	0.13	50796.0	0.24	50741	0.13	50673
X-n586-k159	190484.1	0.09	190405	0.05	190442.2	0.07	190395	0.04	190316
X-n599-k92	108652.1	0.19	108586	0.12	108619.3	0.16	108553	0.09	108451
X-n613-k62	59660.3	0.21	59627	0.15	59619.3	0.14	59558	0.04	59535
X-n627-k43	62271.1	0.17	62242	0.13	62270.4	0.17	62242	0.13	62164
X-n641-k35	63830.3	0.23	63741	0.09	63830.3	0.23	63763	0.12	63684
X-n655-k131	106808.1	0.03	106802	0.02	106794.7	0.01	**106780**	0.00	106780
X-n670-k130	146732.1	0.27	146624	0.20	146553.4	0.15	146452	0.08	146332
X-n685-k75	68320.8	0.17	68273	0.10	68313.1	0.16	68243	0.06	68205
X-n701-k44	82121.0	0.24	82032	0.13	82121.0	0.24	82058	0.16	81923
X-n716-k35	43481.1	0.25	43451	0.18	43474.1	0.23	43440	0.15	43373
X-n733-k159	136389.9	0.15	136314	0.09	136333.9	0.11	136267	0.06	136187
X-n749-k98	77532.1	0.34	77350	0.10	77532.1	0.34	77350	0.10	77269

Continuation of Table 24

Instance	ICAHGS				DPIGA-HGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
X-n766-k71	114719.0	0.26	114646	0.20	114713.4	0.26	114670	0.22	114417
X-n783-k48	72736.2	0.48	72619	0.32	72668.8	0.39	72619	0.32	72386
X-n801-k40	73450.1	0.20	73393	0.12	73450.1	0.20	73380	0.10	73305
X-n819-k171	158503.1	0.24	158368	0.16	158384.7	0.17	158330	0.13	158121
X-n837-k142	194252.2	0.27	194087	0.18	194171.6	0.22	194087	0.18	193737
X-n856-k95	89024.7	0.07	88990	0.03	89017.2	0.06	88983	0.02	88965
X-n876-k59	99675.2	0.38	99521	0.22	99651.8	0.36	99521	0.22	99299
X-n895-k37	54070.5	0.39	53997	0.25	54041.7	0.34	53966	0.20	53860
X-n916-k207	329830.3	0.20	329671	0.15	329830.3	0.20	329671	0.15	329179
X-n936-k151	133085.7	0.28	133047	0.25	133056.9	0.26	133021	0.23	132715
X-n957-k87	85545.3	0.09	85513	0.06	85520.5	0.06	85502	0.04	85465
X-n979-k58	119244.2	0.23	119179	0.17	119245.9	0.23	119179	0.17	118976
X-n1001-k43	72651.5	0.41	72569	0.30	72648.7	0.41	72569	0.30	72355
Min Gap		**0.00**		**0.00**		**0.00**		**0.00**	
Avg. Gap		0.08		0.05		**0.07**		**0.04**	
Max Gap		0.48		0.32		**0.41**		0.32	
No. BKS		34		51		**38**		**56**	

5.4.5.1 Evolutionary performance of DPIGA-HGS and ICAHGS

Figure 25 presents a comparison of the Best performance between DPIGA-HGS and ICAHGS over time, with the application of the same compiler and seed numbers to each algorithm. The results clearly indicate that DPIGA-HGS outperforms ICAHGS in terms of performance.

5.4.6 Real-world application

The primary objective of VRP algorithms is to solve real-world problems effectively. To this end, the performance of DPIGA-HGS is compared to that of ICAHGS on 6 instances of the LoggiBUD benchmark (LOGGI. . . ,). Table 25 presents the average and best results obtained at T_{max} from 10 independent runs. Furthermore, Table 26 offers a concise overview

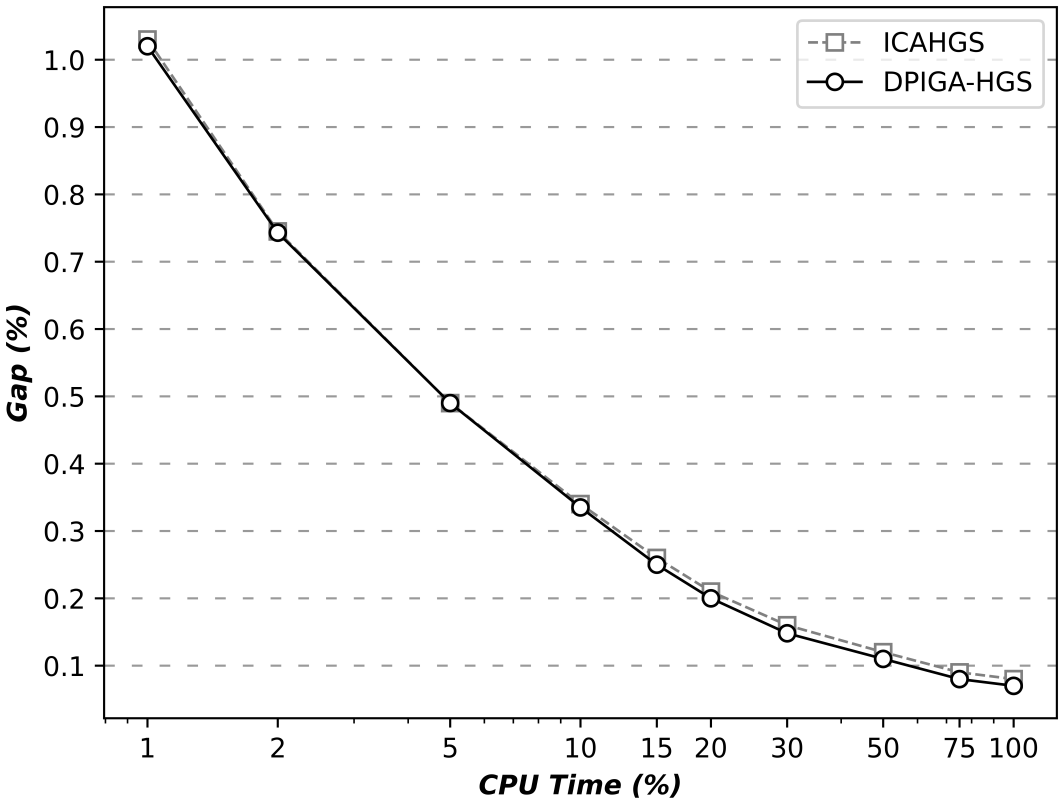

Figure 25 – Performance of DPIGA-HGS and ICAHGS over time

of the gaps in average solution values for both DPIGA-HGS and ICAHGS, along with the number of instances where BKS were achieved.

The results clearly demonstrate the superiority of DPIGA-HGS over ICAHGS, as evidenced by its better performance in all 6 instances concerning average solutions and in 3 out of 6 instances for best solutions, as indicated by the green cells in Table 25. Notably, DPIGA-HGS even discovered a new BKS for Loggi-n901-k42, boasting an impressive -0.03% gap when compared to its previously found BKS. These outcomes underscore the effectiveness and capability of DPIGA-HGS in tackling the challenges posed by real-world VRP instances, further solidifying its status as a promising optimization algorithm for practical applications.

Table 25 – Comparative results ICAHGS, DPIGA-HGS, and BKS for Loggi Benchmark for Urban Deliveries

Instance	ICAHGS				DPIGA-HGS				BKS
	Avg	Gap	Best	Gap	Avg	Gap	Best	Gap	
Loggi-n401-k23	337153.20	0.06	337006	0.02	337148.40	0.06	337006	0.02	336946
Loggi-n501-k24	177505.00	0.02	177469	0.00	177501.60	0.02	177469	0.00	177466
Loggi-n601-k19	113270.20	0.10	113220	0.06	113248.70	0.08	113179	0.02	113155
Loggi-n601-k42	347081.20	0.01	**347059**	0.00	347064.30	0.00	**347059**	0.00	347059
Loggi-n901-k42	246661.80	0.10	246523	0.04	246541.60	0.05	**246335**	-0.03	246418
Loggi-n1001-k31	285889.60	0.54	285649	0.45	285630.50	0.45	285094	0.26	284356

Table 26 – Summary of the results ICAHGS and DPIGA-HGS on 6 instances of LoggiBUD benchmark

	ICAHGS	DPIGA-HGS
Min Gap	0.01	0.01
Avg Gap	0.14	**0.11**
Max Gap	0.54	**0.45**
Number of BKS	1	**2**

5.4.7 Statistical analysis

The performance of the DPIGA-HGS algorithm was evaluated in comparison to ICAHGS by conducting both t-tests and Wilcoxon tests on the average of solutions obtained from all the tested benchmarks (refer to section 4.4.2). The results are presented in Table 27. It was observed that the DPIGA-HGS algorithm demonstrated significantly superior performance compared to ICAHGS on the Uchoa benchmark, with $\rho < 0.05$. However, for the Golden, LoggiBUD, and CMT benchmarks, no significant performance differences were found between the two algorithms.

Table 27 – Statistical results between DPIGA-HGS and ICAHGS on different benchmarks

	p-values	
Benchmarks	**t-test**	**Wilcoxon**
Uchoa	<.001	<.001
Golden	.736	.852
LoggiBUD	.998	.999
CMT	Nan	Nan

6 DISCUSSION AND CONCLUSION

6.1 Discussion

The CVRP is an important problem in the operational level of the supply chain management, dealing with transportation plans and routing decisions. At the strategic level, one has to deal with high level decisions related to the fleet size, operating with own or leased fleet, defining the number and location of factories, facilities or distribution centers, usually named as depots in the CVRP and multi-depot CVRP formulation.

In this study, two hybrid methods are developed to improve the quality of the solutions for the CVRP problem and obtained results confirm this claim. CVRP is one of the basic problems in the VRP domain. To make the CVRP more simplified, lots of constraints are relaxed while they cannot be ignored in a real-world application; such as fixed number of vehicles, distance between nodes which is calculated based on Euclidian distance, hence ignoring traffic congestions, fixed capacity and cost for all the vehicles and so on. All these constraints are the CVRP limitations when it comes to a real-case application.

Nonetheless, despite the mentioned limitations, solving a standard CVRP optimally or near to optimal might have a great impact on providing solutions for real cases based on the CVRP. Since applications based on the CVRP are normally executed at operational levels, the present research can play significant role in cost and time savings. Variants of the standard setting can incorporate problem-specific constraints to the problem and the proposed method can be easily adapted to these situations simply by making changes in the fitness function. All the algorithm steps can be preserved with none or little modifications. Depending on the application, a well suited VRP variant can be adopted while its implementation might be affected by complexity of the proposed solution. It is worth noting that most of the VRP variants are inherited from the CVRP by considering different constraints and initial conditions.

6.2 Conclusion

In this study, at first a hybrid algorithm (ICAHGS) for solving CVRP is proposed, combining a refined ICA and HGS-CVRP. As stated, HGS-CVRP has many advantages that come from its population management, diversity control and improved local search. On the other hand, ICA has a great ability in global search to find higher quality solutions. Thus, their combination can improve local search and global search. The experimental results on the Uchoa benchmark confirm that the combination of these algorithms results in a more

powerful algorithm for CVRP. Further, where the multi-restart mechanism refinement to ICA was unable to improve performance, in the instances where both ICAHGS and HGS-CVRP achieved BKS, improved efficiency was achieved. Comparison with other state-of-the-art algorithms confirmed ICAHGS superiority. Further experiments on the CMT and Golden benchmarks provided for increased generality, confirming performance improvement on other datasets. Finally, real-world testing on the LoggiBUD benchmark further confirmed the enhanced performance achieved with ICAHGS.

In chapter 5 the second approach, Dynamic Population Island GA and HGS (DPIGA-HGS) is introduced, which is a novel hybrid metaheuristic model. DPIGA-HGS integrates a specialized island model (DPIGA) and a refined HGS as its local search engine within each island. The primary objective of DPIGA-HGS is to contribute to the advancement of the field by proposing a new variant of Island GA and simultaneously achieving improved optimization results in comparison to ICAHGS. The DPIGA-HGS algorithm has proven to be a highly effective and robust optimization approach in tackling a diverse set of benchmark instances. Through an extensive evaluation process, which encompassed a wide range of real-world problem scenarios, DPIGA-HGS demonstrated its ability to consistently yield high-quality solutions. The results of the comparative analyses revealed the superior performance of DPIGA-HGS when pitted against other state-of-the-art algorithms, including ICAHGS. Across multiple benchmark datasets, DPIGA-HGS showcased its prowess by achieving a significant number of BKS, outperforming its competitors in various instances. Furthermore, DPIGA-HGS exhibited remarkable adaptability across different problem sizes. For smaller instances, it achieved optimal or near-optimal solutions, while maintaining competitive performance on larger instances, as reflected in the maximum and average gap improvements. Moreover, the application of the proposed multi-step restart mechanism (MSRM) further enhanced the algorithm's efficiency, leading to the discovery of additional BKS in several instances. Statistical analyses, including the t-test and Wilcoxon test, corroborated the algorithm's superior performance on specific benchmark datasets, further validating its robustness and efficiency.

6.3 Published articles

This study has resulted in the publication of an article in the "Applied Soft Computing" journal, while the second article is currently under review. The published article is as follows:

- Babak Rezaei, Frederico Gadelha Guimaraes, Rasul Enayatifar, Pauline C. Haddow, Combining genetic local search into a multi-population Imperialist Competitive Algorithm for the Capacitated Vehicle Routing Problem, Applied Soft Computing, 2023, 110309, ISSN 1568-4946, ⟨https://doi.org/10.1016/j.asoc.2023.110309⟩.

6.4 Future directions

As this research has demonstrated the effectiveness of ICAHGS and DPIGA-HGS in addressing various aspects of the Capacitated Vehicle Routing Problem (CVRP), it also highlights several promising avenues for future research and development in the field of metaheuristic optimization and logistics. The following future directions provide a roadmap for enhancing the capabilities and applicability of these algorithms:

1. **Extension to other VRP variants**

   - VRP with Time Windows (VRPTW): Extend the applicability of ICAHGS and DPIGA-HGS to solve the Vehicle Routing Problem with Time Windows (VRPTW). Investigate modifications and adaptations required to handle constraints related to time windows, where each customer must be served within a specified time frame. Explore the integration of time window constraints into the algorithms' solution representation and optimization process. Conduct comprehensive experiments to evaluate the algorithms' performance on VRPTW instances.
   - Pickup and Delivery VRP (PDVRP): Adapt ICAHGS and DPIGA-HGS to address the Pickup and Delivery Vehicle Routing Problem (PDVRP). PDVRP involves the transport of goods where each delivery is associated with a corresponding pickup location. Develop solution representations and operators that account for pickup and delivery pairs. Assess the algorithms' ability to optimize routes considering both pickup and delivery constraints.

2. **Application to VRP with Drones (VRPD)**

   - Integration of drones: Explore the integration of unmanned aerial vehicles (drones) into the routing process to address the Vehicle Routing Problem with Drones (VRPD). Investigate how ICAHGS and DPIGA-HGS can be extended to optimize routes that include drone-assisted deliveries. Develop algorithms that determine when and where drones should be deployed to improve delivery efficiency, reduce costs, and enhance service quality.
   - Hybrid optimization: Combine the strengths of ICAHGS and DPIGA-HGS with specialized algorithms designed for VRPD. Investigate hybrid optimization approaches that leverage the algorithms' capabilities to optimize ground vehicle routes while concurrently optimizing drone routes. This integration can lead to innovative solutions that harness the advantages of both ground and aerial transportation.

3. **Algorithmic enhancements**

- Parameter tuning: Further investigate parameter tuning strategies to optimize the performance of both ICAHGS and DPIGA-HGS. Explore the effects of different parameter settings on convergence, computational efficiency, and solution quality. Employ advanced optimization techniques, such as auto-tuning algorithms or machine learning-based approaches, to automatically adjust parameters during runtime.

4. **Scalability and Handling Large Instances**

   - Parallelization: Investigate techniques for parallelizing ICAHGS and DPIGA-HGS to efficiently solve larger problem instances. Leverage multi-core processors or distributed computing resources to improve scalability. Develop load-balancing mechanisms to ensure efficient resource utilization in parallel implementations.

5. **Innovative migration policies**

   - Dynamic migration: Design dynamic migration policies that adapt based on island performance, solution quality, or other relevant metrics. Explore strategies for varying migration frequencies, sizes, or topologies during the optimization process. Investigate how dynamic migration can improve convergence speed and solution quality.

REFERENCES

ABRAHAM, A.; NEDJAH, N.; MOURELLE, L. d. M. Evolutionary computation: from genetic algorithms to genetic programming. *Genetic Systems Programming: Theory and Experiences*, p. 1–20, 2006. ISSN 3540298495.

AKPINAR, S. Hybrid large neighbourhood search algorithm for capacitated vehicle routing problem. *Expert Systems with Applications*, v. 61, p. 28–38, 2016. ISSN 0957-4174. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0957417416302482⟩.

ARNOLD, F.; SöRENSEN, K. What makes a vrp solution good? the generation of problem-specific knowledge for heuristics. *Computers Operations Research*, v. 106, p. 280–288, 2019. ISSN 0305-0548. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0305054818300315⟩.

ATASHPAZ-GARGARI, E.; LUCAS, C. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In: *2007 IEEE Congress on Evolutionary Computation*. [S.l.: s.n.]. p. 4661–4667. ISBN 1941-0026.

AUGERAT, P. et al. Computational results with a branch and cut code for the capacitated vehicle routing problem. 1995.

AYOUGH, A.; ZANDIEH, M.; FARSIJANI, H. Ga and ica approaches to job rotation scheduling problem: considering employee's boredom. *The International Journal of Advanced Manufacturing Technology*, v. 60, n. 5, p. 651–666, 2012. ISSN 1433-3015.

BAKER, B. M.; AYECHEW, M. A. A genetic algorithm for the vehicle routing problem. *Computers Operations Research*, v. 30, n. 5, p. 787–800, 2003. ISSN 0305-0548.

BALDACCI, R.; CHRISTOFIDES, N.; MINGOZZI, A. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, v. 115, p. 351–385, 2008. ISSN 0025-5610.

BALINSKI, M. L.; QUANDT, R. E. On an integer program for a delivery problem. *Operations research*, v. 12, n. 2, p. 300–304, 1964. ISSN 0030-364X.

BARBAROSOGLU, G.; OZGUR, D. A tabu search algorithm for the vehicle routing problem. *Computers Operations Research*, v. 26, n. 3, p. 255–270, 1999. ISSN 0305-0548.

BAUM, E. B. Towards practical 'neural'computation for combinatorial optimization problems. In: . [S.l.]: American Institute of Physics. v. 151, p. 53–58. ISBN 088318351X.

BEASLEY, J. E. Route first—cluster second methods for vehicle routing. *Omega*, v. 11, n. 4, p. 403–408, 1983. ISSN 0305-0483.

BELENGUER, J.-M. et al. Split-delivery capacitated arc-routing problem: Lower bound and metaheuristic. *Transportation Science*, v. 44, n. 2, p. 206–220, 2010. ISSN 0041-1655.

BERGER, J.; BARKAOUI, M. A new hybrid genetic algorithm for the capacitated vehicle routing problem. *Journal of the operational Research Society*, v. 54, p. 1254–1262, 2003. ISSN 0160-5682.

BEULLENS, P. et al. A guided local search heuristic for the capacitated arc routing problem. *European Journal of Operational Research*, v. 147, n. 3, p. 629–643, 2003. ISSN 0377-2217.

BODENHOFER, U. Genetic algorithms: theory and applications. *Lecture notes, Fuzzy Logic Laboratorium Linz-Hagenberg, Winter*, v. 2004, 2003.

BOUDIA, M.; PRINS, C. A memetic algorithm with dynamic population management for an integrated production–distribution problem. *European Journal of Operational Research*, v. 195, n. 3, p. 703–715, 2009. ISSN 0377-2217.

BRANDãO, J. A tabu search algorithm for the heterogeneous fixed fleet vehicle routing problem. *Computers Operations Research*, v. 38, n. 1, p. 140–151, 2011. ISSN 0305-0548.

BRANKE J., K. A. . S. H. Distribution of evolutionary algorithms in heterogeneous networks. In: *Genetic and Evolutionary Computation–GECCO 2004: Genetic and Evolutionary Computation*. [S.l.]: Springer Berlin Heidelberg. p. 923–934.

BULLNHEIMER, B.; HARTL, R. F.; STRAUSS, C. An improved ant system algorithm for the vehicle routing problem. *Annals of operations research*, v. 89, p. 319–328, 1999. ISSN 0254-5330.

CANTú-PAZ, E. Migration policies, selection pressure, and parallel evolutionary algorithms. *Journal of heuristics*, v. 7, p. 311– 334, 2001. - Migration policies, selection pressure, and parallel evolutionary algorithms - 311.

CHEN, A.-l.; YANG, G.-k.; WU, Z.-m. Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem. *Journal of Zhejiang University-Science A*, v. 7, n. 4, p. 607–614, 2006. ISSN 1673-565X.

CHEN, P.; HUANG, H.-k.; DONG, X.-Y. Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem. *Expert Systems with Applications*, v. 37, n. 2, p. 1620–1627, 2010. ISSN 0957-4174.

CHRISTIAENS, J.; BERGHE, G. V. Slack induction by string removals for vehicle routing problems. *Transportation Science*, v. 54, n. 2, p. 417–433, 2020. ISSN 0041-1655.

CHRISTOFIDES, N. The vehicle routing problem. *Combinatorial optimization*, p. 315–318, 1979.

CHRISTOFIDES, N.; EILON, S. An algorithm for the vehicle-dispatching problem. *Journal of the Operational Research Society*, v. 20, n. 3, p. 309–318, 1969. ISSN 0160-5682.

CLARKE, G.; WRIGHT, J. W. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, v. 12, n. 4, p. 568–581, 1964. ISSN 0030-364X.

CORDEAU, J.-F.; LAPORTE, G.; MERCIER, A. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational research society*, v. 52, n. 8, p. 928–936, 2001. ISSN 0160-5682.

Web Page, CPU Benchmark. Disponível em: ⟨https://www.cpubenchmark.net/compare/Intel-i5-4590T-vs-Intel-Xeon-Gold-6148⟩.

Web Page, CVRPLIB website. Disponível em: ⟨http://vrp.atd-lab.inf.puc-rio.br/index.php/en/⟩.

DANTZIG, G. B.; RAMSER, J. H. The truck dispatching problem. *Management Science*, v. 6, n. 1, p. 80–91, 1959. ISSN 0025-1909. Doi: 10.1287/mnsc.6.1.80.

Web Page, DIMACS. Disponível em: ⟨http://dimacs.rutgers.edu/programs/challenge/vrp/cvrp/⟩.

DORIGO, M.; STUTZLE, T. Ant colony optimization. mit press, cambridge, ma. 2004.

ENAYATIFAR, R.; ABDULLAH, A. H.; LEE, M. A weighted discrete imperialist competitive algorithm (wdica) combined with chaotic map for image encryption. *Optics and Lasers in Engineering*, v. 51, n. 9, p. 1066–1077, 2013. ISSN 0143-8166. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0143816613001048⟩.

FALLAHI, A. E.; PRINS, C.; CALVO, R. W. A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. *Computers Operations Research*, v. 35, n. 5, p. 1725–1741, 2008. ISSN 0305-0548.

FEO, T. A.; RESENDE, M. G. C. A probabilistic heuristic for a computationally difficult set covering problem. *Operations research letters*, v. 8, n. 2, p. 67–71, 1989. ISSN 0167-6377.

FISHER, M. L.; JAIKUMAR, R. A generalized assignment heuristic for vehicle routing. *Networks*, v. 11, n. 2, p. 109–124, 1981. ISSN 0028-3045.

FLESZAR, K.; OSMAN, I. H.; HINDI, K. S. A variable neighbourhood search algorithm for the open vehicle routing problem. *European Journal of Operational Research*, v. 195, n. 3, p. 803–809, 2009. ISSN 0377-2217.

FUKASAWA, R. et al. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical programming*, v. 106, p. 491–511, 2006. ISSN 0025-5610.

GILLETT, B. E.; MILLER, L. R. A heuristic algorithm for the vehicle-dispatch problem. *Operations Research*, v. 22, n. 2, p. 340–349, 1974. ISSN 0030-364X.

GLOVER, F. Heuristics for integer programming using surrogate constraints. *Decision sciences*, v. 8, n. 1, p. 156–166, 1977. ISSN 0011-7315.

GLOVER, F. Future paths for integer programming and links to artificial intelligence. *Computers operations research*, v. 13, n. 5, p. 533–549, 1986. ISSN 0305-0548.

GLOVER, F. Tabu search—part i. *ORSA Journal on computing*, v. 1, n. 3, p. 190–206, 1989. ISSN 0899-1499.

GLOVER, F. Tabu search—part ii. *ORSA Journal on computing*, v. 2, n. 1, p. 4–32, 1990. ISSN 0899-1499.

GLOVER, F. Tabu search and adaptive memory programming — advances, applications and challenges. In: _____. *Interfaces in Computer Science and Operations Research: Advances in Metaheuristics, Optimization, and Stochastic Modeling Technologies*. Boston, MA: Springer US, 1997. p. 1–75. ISBN 978-1-4615-4102-8. Disponível em: ⟨https://doi.org/10.1007/978-1-4615-4102-8_1⟩.

GLOVER, F.; LAGUNA, M.; MARTI, R. Fundamentals of scatter search and path relinking. *Control and Cybernetics*, Vol. 29, no 3, p. 653–684, 2000.

GOLDBERG D. E., . H. J. H. Book. *Genetic algorithms and machine learning*. [S.l.: s.n.], 1989.

GOLDEN, B. L. et al. Generic, *Metaheuristics in vehicle routing, Fleet management and logistics, TG Crainic and G. Laporte*. [S.l.]: Kluwer, Boston, 1998.

Web Page, GOOGLE OR-Tools. Disponível em: ⟨https://developers.google.com/optimization/routing⟩.

GROëR, C.; GOLDEN, B.; WASIL, E. A parallel algorithm for the vehicle routing problem. *INFORMS Journal on Computing*, v. 23, n. 2, p. 315–330, 2011. ISSN 1091-9856.

GULCZYNSKI, D.; GOLDEN, B.; WASIL, E. The period vehicle routing problem: New heuristics and real-world variants. *Transportation Research Part E: Logistics and Transportation Review*, v. 47, n. 5, p. 648–668, 2011. ISSN 1366-5545.

HERTZ, A.; MITTAZ, M. A variable neighborhood descent algorithm for the undirected capacitated arc routing problem. *Transportation science*, v. 35, n. 4, p. 425–434, 2001. ISSN 0041-1655.

HO, S. C.; GENDREAU, M. Path relinking for the vehicle routing problem. *Journal of Heuristics*, v. 12, n. 1, p. 55–72, 2006. ISSN 1572-9397. Disponível em: ⟨https://doi.org/10.1007/s10732-006-4192-1⟩.

HOLLAND, J. H. *Adaptation in natural and artificial systems.* [S.l.]: Cambridge. MA: MIT Press, 1973.

IQBAL, S.; KAYKOBAD, M.; RAHMAN, M. S. Solving the multi-objective vehicle routing problem with soft time windows with the help of bees. *Swarm and evolutionary computation*, v. 24, p. 50–64, 2015. ISSN 2210-6502.

KENNEDY, J. Particle swarm optimization/kennedy j., eberhart rc. In: . [S.l.: s.n.]. p. 12–13.

KENNEDY, J.; EBERHART, R. C. A discrete binary version of the particle swarm algorithm. In: . [S.l.]: IEEE. v. 5, p. 4104–4108. ISBN 0780340531.

KILBY, P.; PROSSER, P.; SHAW, P. Guided local search for the vehicle routing problem with time windows. *Meta-heuristics: Advances and trends in local search paradigms for optimization*, p. 473–486, 1999. ISSN 1461376467.

KIRKPATRICK, S.; JR, C. D. G.; VECCHI, M. P. Optimization by simulated annealing. *science*, v. 220, n. 4598, p. 671–680, 1983. ISSN 0036-8075.

KUO, Y.; WANG, C.-C. A variable neighborhood search for the multi-depot vehicle routing problem with loading cost. *Expert Systems with Applications*, v. 39, n. 8, p. 6949–6954, 2012. ISSN 0957-4174.

KYTöJOKI, J. et al. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers operations research*, v. 34, n. 9, p. 2743–2757, 2007. ISSN 0305-0548.

LACOMME, P.; PRINS, C.; RAMDANE-CHERIF, W. Competitive memetic algorithms for arc routing problems. *Annals of Operations Research*, v. 131, p. 159–185, 2004. ISSN 0254-5330.

LETCHFORD, A. N.; EGLESE, R. W.; LYSGAARD, J. Multistars, partial multistars and the capacitated vehicle routing problem. *Mathematical Programming*, v. 94, p. 21–40, 2002. ISSN 0025-5610.

LEUNG, S. C. H. et al. A meta-heuristic algorithm for heterogeneous fleet vehicle routing problems with two-dimensional loading constraints. *European Journal of Operational Research*, v. 225, n. 2, p. 199–210, 2013. ISSN 0377-2217.

LEUNG, S. C. H. et al. Extended guided tabu search and a new packing algorithm for the two-dimensional loading vehicle routing problem. *Computers Operations Research*, v. 38, n. 1, p. 205–215, 2011. ISSN 0305-0548.

LI, F.; GOLDEN, B.; WASIL, E. Very large-scale vehicle routing: new test problems, algorithms, and results. *Computers Operations Research*, v. 32, n. 5, p. 1165–1179, 2005. ISSN 0305-0548.

LIN, C. et al. Survey of green vehicle routing problem: past and future trends. *Expert systems with applications*, v. 41, n. 4, p. 1118–1138, 2014. ISSN 0957-4174.

LIN, S. Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, v. 44, n. 10, p. 2245–2269, 1965. ISSN 0005-8580.

LIN, S.; KERNIGHAN, B. W. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, v. 21, n. 2, p. 498–516, 1973. ISSN 0030-364X.

LIN, S.-W. Solving the team orienteering problem using effective multi-start simulated annealing. *Applied Soft Computing*, v. 13, n. 2, p. 1064–1073, 2013. ISSN 1568-4946.

LIN, S.-W.; VINCENT, F. Y. A simulated annealing heuristic for the team orienteering problem with time windows. *European Journal of Operational Research*, v. 217, n. 1, p. 94–107, 2012. ISSN 0377-2217.

LIU, S.-C.; CHEN, A.-Z. Variable neighborhood search for the inventory routing and scheduling problem in a supply chain. *Expert Systems with Applications*, v. 39, n. 4, p. 4149–4159, 2012. ISSN 0957-4174.

Web Page, LOGGI Benchmark for Urban Deliveries. Disponível em: ⟨https://github.com/loggi/loggibud⟩.

LOURENçO, H. R.; MARTIN, O. C.; STüTZLE, T. Generic, *Iterated Local Search: Framework and Applications. M. Gendreau, J.-Y. Potvin, eds., Handbook of Metaheuristics, International Series in Operations Research Management Science, vol. 146*. [S.l.]: Springer, Boston, MA, 2010.

LYSGAARD, J.; LETCHFORD, A. N.; EGLESE, R. W. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, v. 100, n. 2, p. 423–445, 2004. ISSN 1436-4646.

MARINAKI, M.; MARINAKIS, Y. A glowworm swarm optimization algorithm for the vehicle routing problem with stochastic demands. *Expert Systems with Applications*, v. 46, p. 145–163, 2016. ISSN 0957-4174.

MARINAKIS, Y. Multiple phase neighborhood search-grasp for the capacitated vehicle routing problem. *Expert Systems with Applications*, v. 39, n. 8, p. 6807–6815, 2012. ISSN 0957-4174.

MARINAKIS, Y.; IORDANIDOU, G.-R.; MARINAKI, M. Particle swarm optimization for the vehicle routing problem with stochastic demands. *Applied Soft Computing*, v. 13, n. 4, p. 1693–1704, 2013. ISSN 1568-4946.

MARINAKIS, Y.; MARINAKI, M. A hybrid multi-swarm particle swarm optimization algorithm for the probabilistic traveling salesman problem. *Computers Operations Research*, v. 37, n. 3, p. 432–442, 2010. ISSN 0305-0548.

MARINAKIS, Y.; MARINAKI, M. A bumble bees mating optimization algorithm for the open vehicle routing problem. *Swarm and Evolutionary Computation*, v. 15, p. 80–94, 2014. ISSN 2210-6502.

Web Page, MATLAB implementation of ICA. Disponível em: ⟨https://www.mathworks.com/matlabcentral/fileexchange/22046-imperialist-competitive-algorithm-ica⟩.

MIRANDA, D. M.; CONCEIçãO, S. V. The vehicle routing problem with hard time windows and stochastic travel and service time. *Expert Systems with Applications*, v. 64, p. 104–116, 2016. ISSN 0957-4174.

MITCHELL, M. Genetic algorithms: An overview. In: CITESEER. *Complex.* [S.l.], 1995. v. 1, n. 1, p. 31–39.

MLADENOVIć, N.; HANSEN, P. Variable neighborhood search. *Computers operations research*, v. 24, n. 11, p. 1097–1100, 1997. ISSN 0305-0548.

MOGHADDAM, B. F.; RUIZ, R.; SADJADI, S. J. Vehicle routing problem with uncertain demands: An advanced particle swarm algorithm. *Computers Industrial Engineering*, v. 62, n. 1, p. 306–317, 2012. ISSN 0360-8352.

MOLE, R. H.; JAMESON, S. R. A sequential route-building algorithm employing a generalised savings criterion. *Journal of the Operational Research Society*, v. 27, n. 2, p. 503–511, 1976. ISSN 0160-5682.

MONTANé, F. A. T.; GALVAO, R. D. A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. *Computers Operations Research*, v. 33, n. 3, p. 595–619, 2006. ISSN 0305-0548.

MONTOYA-TORRES, J. R. et al. A literature review on the vehicle routing problem with multiple depots. *Computers Industrial Engineering*, v. 79, p. 115–129, 2015. ISSN 0360-8352.

MOSCATO, P. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. *Caltech concurrent computation program, C3P Report*, v. 826, n. 1989, p. 37, 1989.

MOSCATO, P. Memetic algorithms: A short introduction. *New ideas in optimization*, p. 219–234, 1999.

NAGATA, Y.; BRäYSY, O. Edge assembly-based memetic algorithm for the capacitated vehicle routing problem. *Networks: An International Journal*, v. 54, n. 4, p. 205–215, 2009. ISSN 0028-3045.

NGUEVEU, S. U.; PRINS, C.; CALVO, R. W. An effective memetic algorithm for the cumulative capacitated vehicle routing problem. *Computers Operations Research*, v. 37, n. 11, p. 1877–1885, 2010. ISSN 0305-0548.

NGUYEN, V.-P.; PRINS, C.; PRODHON, C. Solving the two-echelon location routing problem by a grasp reinforced by a learning process and path relinking. *European Journal of Operational Research*, v. 216, n. 1, p. 113–126, 2012. ISSN 0377-2217.

OLIVER, I. M.; SMITH, D.; HOLLAND, J. R. C. Study of permutation crossover operators on the traveling salesman problem. In: . [S.l.]: Hillsdale, NJ: L. Erlhaum Associates, 1987. p. 224–230.

OR, I. Book. *TRAVELING SALESMAN TYPE COMBINATORIAL PROBLEMS AND THEIR RELATION TO THE LOGISTICS OF REGIONAL BLOOD BANKING*. [S.l.]: Northwestern University, 1976. ISBN 9798660392177.

OSMAN, I. H. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of operations research*, v. 41, p. 421–451, 1993. ISSN 0254-5330.

PARRAGH, S. N.; DOERNER, K. F.; HARTL, R. F. A survey on pickup and delivery models part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, v. 58, p. 81–117, 2006.

PARRAGH, S. N.; DOERNER, K. F.; HARTL, R. F. A survey on pickup and delivery problems: Part i: Transportation between customers and depot. *Journal für Betriebswirtschaft*, v. 58, p. 21–51, 2008. ISSN 0344-9327.

PECIN, D. et al. Improved branch-cut-and-price for capacitated vehicle routing. *Mathematical Programming Computation*, v. 9, p. 61–100, 2017. ISSN 1867-2949.

PISINGER, D.; ROPKE, S. A general heuristic for vehicle routing problems. *Computers operations research*, v. 34, n. 8, p. 2403–2435, 2007. ISSN 0305-0548.

POLACEK, M. et al. A variable neighborhood search for the capacitated arc routing problem with intermediate facilities. *Journal of Heuristics*, v. 14, p. 405–423, 2008. ISSN 1381-1231.

POPOVIć, D.; VIDOVIć, M.; RADIVOJEVIć, G. Variable neighborhood search heuristic for the inventory routing problem in fuel delivery. *Expert Systems with Applications*, v. 39, n. 18, p. 13390–13398, 2012. ISSN 0957-4174.

POTVIN, J.-Y.; BENGIO, S. The vehicle routing problem with time windows part ii: genetic search. *INFORMS journal on Computing*, v. 8, n. 2, p. 165–172, 1996. ISSN 1091-9856.

PRINS, C. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers Operations Research*, v. 31, n. 12, p. 1985–2002, 2004. ISSN 0305-0548. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0305054803001588⟩.

PRINS, C.; LABADI, N.; REGHIOUI, M. Tour splitting algorithms for vehicle routing problems. *International Journal of Production Research*, v. 47, n. 2, p. 507–535, 2009. ISSN 0020-7543.

PRINS, C.; PRODHON, C.; CALVO, R. W. Solving the capacitated location-routing problem by a grasp complemented by a learning process and a path relinking. *4OR*, v. 4, p. 221–238, 2006. ISSN 1619-4500.

PRINS, C.; SEVAUX, M.; SÖRENSEN, K. A genetic algorithm with population management (ga— pm) for the carp. In: *Tristan V (5th Triennal Symposium on Transportation Analysis*. [S.l.: s.n.], 2004.

PRODHON, C.; PRINS, C. A memetic algorithm with population management (ma—pm) for the periodic location-routing problem. In: . [S.l.]: Springer. p. 43–57. ISBN 3540884386.

QU, Y.; BARD, J. F. A grasp with adaptive large neighborhood search for pickup and delivery problems with transshipment. *Computers Operations Research*, v. 39, n. 10, p. 2439–2456, 2012. ISSN 0305-0548.

REEVES, C. R. Chapter 3: Genetic algorithms. in: Handbook of metaheuristics, international series in operations research and management science. michel gendreau and jean-yves potvin (eds). *Handbook of metaheuristics*, p. 109–139, 2010. ISSN 1441916636.

REGO, C.; ROUCAIROL, C. A parallel tabu search algorithm using ejection chains for the vehicle routing problem. *Meta-heuristics: Theory and applications*, p. 661–675, 1996. ISSN 1461285879.

REIMANN, M.; DOERNER, K.; HARTL, R. F. D-ants: Savings based ants divide and conquer the vehicle routing problem. *Computers Operations Research*, v. 31, n. 4, p. 563–591, 2004. ISSN 0305-0548.

ROBINSON, J. *On the Hamiltonian game (a traveling salesman problem)*. [S.l.]: Rand Corporation, 1949.

RUSSELL, R. A.; CHIANG, W.-C. Scatter search for the vehicle routing problem with time windows. *European Journal of Operational Research*, v. 169, n. 2, p. 606–622, 2006. ISSN 0377-2217. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0377221704005570⟩.

SADAEI, H. J. et al. A hybrid model based on differential fuzzy logic relationships and imperialist competitive algorithm for stock market forecasting. *Applied Soft Computing*, v. 40, p. 132–149, 2016. ISSN 1568-4946. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S1568494615007449⟩.

SANTOS, L.; COUTINHO-RODRIGUES, J.; CURRENT, J. R. An improved ant colony optimization based algorithm for the capacitated arc routing problem. *Transportation Research Part B: Methodological*, v. 44, n. 2, p. 246–266, 2010. ISSN 0191-2615.

SBAI, I.; KRICHEN, S.; LIMAM, O. Two meta-heuristics for solving the capacitated vehicle routing problem: the case of the tunisian post office. *Operational Research*, v. 22, n. 1, p. 507–549, 2022. ISSN 1866-1505.

SCHMID, V. et al. Hybridization of very large neighborhood search for ready-mixed concrete delivery problems. *Computers operations research*, v. 37, n. 3, p. 559–574, 2010. ISSN 0305-0548.

SHAW, P. Using constraint programming and local search methods to solve vehicle routing problems. In: . [S.l.]: Springer. p. 417–431. ISBN 3540652248.

SILVA, M. M.; SUBRAMANIAN, A.; OCHI, L. S. An iterated local search heuristic for the split delivery vehicle routing problem. *Computers Operations Research*, v. 53, p. 234–249, 2015. ISSN 0305-0548.

STUDENT. The probable error of a mean. *Biometrika*, p. 1–25, 1908. ISSN 0006-3444.

SUBRAMANIAN, A. et al. A hybrid algorithm for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*, v. 221, n. 2, p. 285–295, 2012. ISSN 0377-2217.

SöRENSEN, K.; SEVAUX, M. Ma— pm: memetic algorithms with population management. *Computers Operations Research*, v. 33, n. 5, p. 1214–1225, 2006. ISSN 0305-0548.

TAILLARD, Parallel iterative search methods for vehicle routing problems. *Networks*, v. 23, n. 8, p. 661–673, 1993. ISSN 0028-3045. Https://doi.org/10.1002/net.3230230804. Disponível em: ⟨https://doi.org/10.1002/net.3230230804⟩.

TALBI, E.-G. Book. *Metaheuristics: from design to implementation*. [S.l.]: John Wiley Sons, 2009. v. 74. ISBN 0470496908.

TARANTILIS, C. D.; KIRANOUDIS, C. T. A flexible adaptive memory-based algorithm for real-life transportation operations: Two case studies from dairy and construction sector. *European Journal of Operational Research*, v. 179, n. 3, p. 806–822, 2007. ISSN 0377-2217.

THANGIAH, S. R. Vehicle routing with time windows using genetic algorithms. In: _____. *Practical handbook of genetic algorithms*. [S.l.]: CRC press, 2019. p. 253–278. ISBN 0429128339.

TOTH, P.; VIGO, D. The granular tabu search and its application to the vehicle-routing problem. *Informs Journal on computing*, v. 15, n. 4, p. 333–346, 2003. ISSN 1091-9856.

TOTH, P.; VIGO, D. Book. *Vehicle routing: problems, methods, and applications*. [S.l.]: SIAM, 2014. ISBN 1611973589.

UCHOA, E. et al. New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, v. 257, n. 3, p. 845–858, 2017. ISSN 0377-2217. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0377221716306270⟩.

USBERTI, F. L.; FRANçA, P. M.; FRANçA, A. L. M. Grasp with evolutionary path-relinking for the capacitated arc routing problem. *Computers Operations Research*, v. 40, n. 12, p. 3206–3217, 2013. ISSN 0305-0548.

VANSTEENWEGEN, P. et al. Iterated local search for the team orienteering problem with time windows. *Computers Operations Research*, v. 36, n. 12, p. 3281–3290, 2009. ISSN 0305-0548.

VIDAL, T. Hybrid genetic search for the cvrp: Open-source implementation and swap* neighborhood. *Computers Operations Research*, v. 140, p. 105643, 2022. ISSN 0305-0548. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S030505482100349X⟩.

VIDAL, T. et al. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, v. 60, n. 3, p. 611–624, 2012. ISSN 0030-364X.

VIDAL, T. et al. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers operations research*, v. 40, n. 1, p. 475–489, 2013. ISSN 0305-0548.

VIDAL, T. et al. A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research*, v. 234, n. 3, p. 658–673, 2014. ISSN 0377-2217.

Web Page, VIDAL'S Github repository. Disponível em: ⟨https://github.com/vidalt/HGS-CVRP⟩.

WHITLEY, D. A genetic algorithm tutorial. *Statistics and computing*, v. 4, p. 65–85, 1994. ISSN 0960-3174.

WILCOXON, F. Probability tables for individual comparisons by ranking methods. *Biometrics*, v. 3, n. 3, p. 119–122, 1947. ISSN 0006-341X.

YOUSEFIKHOSHBAKHT, M.; SEDIGHPOUR, M. New imperialist competitive algorithm to solve the travelling salesman problem. *International Journal of Computer Mathematics*, v. 90, n. 7, p. 1495–1505, 2013. ISSN 0020-7160.