Chapter 7
Vehicle Routing Problems

7.1 Introduction

Vehicle routing problems (VRP) [72, 50, 186] represent one of the most investigated combinatorial optimization problems [134], due to the problem complexity and their potential impact on real-world applications especially in logistics and supply chains. The basic VRP involves constructing a set of closed routes from and to a depot, each serviced by a vehicle of certain capacity. In each route, a vehicle delivers the required demand in an ordered list of tasks for customers. The objective is to minimize the total distance, satisfying the capacity for all customers on each route. In some problem variants the number of vehicles used is also minimized. More details of the problem models and benchmark datasets are given in Appendix B.3.

In Operational Research, combinatorial optimization problems (see Appendix B) represent a subset of optimization, which consists of assigning discrete domain values to integer decision variables in a problem. As one of the mostly studied combinatorial optimization problems, the basic VRP is NP-hard [65, 101]. When modelling real-world VRP applications such as routing in transport logistics, the introduction of a wide range of constraints further increases the complexity. Variants of VRP thus have been studied in research integrating common features ranging from time windows, to capabilities to uncertainties of the tasks. Meta-heuristics integrating simple techniques as well as exact methods [185] thus represent one of the recent promising directions addressing both the complexity and realistic features in variants of VRP (see surveys for VRP [97, 186], VRP with Time Windows (VRPTW) [22], Capacitated VRP (CVRP) [68]) and Dynamic VRP (DVRP) [159, 136].

The most studied variants of VRP in HH (Appendix B.3) are reviewed in this chapter to obtain observations and draw conclusions about the performance of Hyper-heuristics (HH) for the problems. At the low level, a range of domain specific *llh*, which are either low-level heuristics in selection HH or components in generation HH, are adaptively selected (Section 7.2). At the high level, both *selection* and *generation* HH (Sections 7.3 and 7.4) have been developed to configure both

N. Pillay, R. Qu, *Hyper-Heuristics: Theory and Applications*,
Natural Computing Series, https://doi.org/10.1007/978-3-319-96514-7_7

types of these *llh*. Due to the generality of HH, some of the research developed HH approaches across different problem domains. Representative work along a line / series of research developments, with focus on VRP, has been reviewed. HH across multiple domains is discussed in Chapter 11.

In the rich literature of VRP research, a large number of VRP variants (VRP, VRPTW, CVRP, and DVRP, etc.) have been modelled, with benchmark datasets established when evaluating meta-heuristics. Although VRP has not been examined extensively in HH as in meta-heuristic algorithms, most of these widely studied benchmark VRP (see Appendix B.3) have also been considered in HH, and in some cases evaluated against meta-heuristics. Interesting observations that emerged are discussed in Section 7.5, leading to new lines of future potential research on real VRP applications with more real constraints or problem features.

7.2 Low-Level Heuristics for Vehicle Routing Problems

In the most commonly used solution encoding in the VRP literature, customers or tasks are usually represented as nodes in a graph representing the routing network. This graph-based representation is used in this chapter. Based on the classifications of HH [31], *llh* used in VRP are grouped into constructive in Section 7.2.1 and perturbative in Section 7.2.2. These *llh*, configured by the high-level heuristics in HH, operate upon nodes, which represent either customers or tasks. Most of these *llh* are usually problem specific, and have been extended or hybridized in HH for VRP variants with different constraints and features.

7.2.1 Constructive Low-Level Heuristics in Vehicle Routing Problems

Two types of constructive *llh* are used in the VRP literature for selection and generation HH, respectively.

- In selection HH, classic constructive heuristics such as the *Saving* heuristic by Clarke and Wright [48] have been employed to construct routes in VRP. In some HH approaches, perturbative heuristics are also used together with these constructive *llh* to further improve the generated solutions; see Section 7.3.
- In generation HH, problem attributes (state attributes or problem features) are used and combined using function operators or grammars to generate new heuristic functions (trees) or sequences of heuristic templates, see Section 7.4. These attributes and operators / grammars can also be seen as elements of constructive *llh*, which are configured by the high-level heuristics in HH to construct solutions for the VRP.

In selection constructive HH, the most commonly used constructive *llh* are summarized as follows. These heuristics and their extensions and hybridizations are widely used in the VRP literature.

- *Greedy*: insert randomly selected customers, subject to constraints, into those routes incurring the minimum cost
- *Saving* by Clarke and Wright [48]: merge two routes into one, based on the savings obtained on the distance of the resulting route
- *Insertion* by Mole and Jameson [123]: insert customers into a route that leads to the minimum cost resulting from the insertion
- *Sweep* by Gillett and Miller [69]: subject to the capacity of vehicles, form clusters of nodes by rotating a ray through (i.e. sweeping) customers clockwise or anticlockwise from the central depot; each route is then built by considering each cluster of nodes as a travelling salesman problem
- *Ordering*: heuristics that order and insert customers into the routes using certain criteria, i.e. increasing / decreasing demand, and farthest / nearest to the depot, etc.

In generation constructive HH, each heuristic is represented as a tree or grammar, with terminals (i.e. problem attributes) combined by function operators (i.e. internal nodes) or grammars. These constructive heuristics are generated either online or offline based on a set of VRP training instances [175, 104], and used to choose nodes in the network when constructing routes for the VRP. The function operators (see Table 7.1) or grammars are usually general across different problems; see also Chapter 9 for packing problems. Terminals (problem attributes) are usually problem dependent; we show below the ones widely used in HH for VRP.

- *demand*: (expected) (normalized) demand of the task for a customer
- *capacity / load*: (normalized) capacity when leaving the pickup or delivery node
- *cost*: cost of delivering the demand for a customer
- *distance*: distance from the current node or the depot; average distance of the current nodes to the remaining nodes; standard deviation of distance to the remaining nodes; and the total distance of routes, etc.
- *time* related: for a node, (normalized) start *stw* and end *etw* of the time window; arrival *at* and departure *dt* times; service time *st*; wait time *wt* at the pickup or delivery node; and time *t* from the current node or depot; etc.
- *satisfied*: proportion of nodes already being served
- *depotCost*: cost to reach the depot from the node
- *most_constrained*: most constrained tasks by demand or capacity
- *angle*: in relation to the *Sweep* heuristic, the angle between the current node and the depot
- *density*: node density in the routing network graph

Table 7.1 Most used function operators in genetic programming

Function	**Operation**
Addition $(+)$	addition, which adds values of two child nodes
Subtraction $(-)$	subtraction, which subtracts values of two child nodes
Multiplication $(*)$	multiplication, which multiplies values of two child nodes
Protected division $(/)$	division, which divides values of two child nodes; when denominator is zero a protected operation is used
Relational operators $(\leq, <, >, \geq, =, \neq)$	comparison, which returns $true$ or $false$ based on the values of two child nodes
(*exp***)**	exponential function, which returns e^x, where x is the value of the child node
(*max***)** or **(***min***)**	returns the maximum or minimum value of two child nodes
(*angle***)**	angle between the coordinates of the node and the origin of the polar system (usually the depot)

7.2.2 Perturbative Low-Level Heuristics in Vehicle Routing Problems

Widely used perturbative operators in meta-heuristics have been used as *llh* in selection HH, in some research hybridized with constructive *llh*, for variants of VRP. In the literature, there is no existing research employing perturbative *llh* in generation HH. In most selection perturbative HH approaches, a subset or extensions of the following perturbative *llh* have been employed. Operations of these *llh* upon the routing solutions are usually subject to the constraints in the VRP.

- *shift*: move a single node to a different route
- *swap*: swap two adjacent nodes in a route
- *interchange*: swap two nodes from different routes
- *or-opt*: move consecutive nodes to a different position in the same route
- λ*-opt*: exchange λ edges in a route
- *Van Breedam* [66]: relocate, exchange and cross strings between two routes
- *crossover*: exchange (part of) routes between two solutions
- *ruin and recreate* (*remove and re-insert*, or *destroy and repair*): remove a number of nodes using some criteria (time or location based), and insert them back into selected routes using heuristics. A new route is opened for nodes that cannot be re-inserted due to constraints.

7.3 Selection Hyper-Heuristics for Vehicle Routing Problems

In the existing selection HH, a given set of perturbative *llh* are usually selected to improve initial complete solutions. In some approaches, both constructive and perturbative *llh* are selected to construct and then improve the VRP solutions within one HH framework.

7.3.1 Selection Hyper-Heuristics Using Perturbative Low-Level Heuristics

A variety of techniques have been employed in HH to select perturbative *llh* for VRP. These range from local search [149, 191], to classifiers [8], to a multi-armed-bandit mechanism [166]. The selected perturbative *llh* (Section 7.2.2), using online or offline learning, or by solution evaluations, are applied to iteratively improve complete solutions for variants of VRP.

Although not named HH, an adaptive large-neighbourhood search at the master (high) level within a unified framework is developed in [149] to address a unified pickup and delivery problem with time windows for five variants of VRP. Using roulette wheel selection, simple *destroy and repair* heuristics (see Section 7.2.2) compete to modify a large number of variables in complete solutions based on the scores adjusted during online learning. The general framework has improved a large number of best results across all five different variants, demonstrating a highly effective and robust approach with little tuning effort for large-scale real-world VRP with mixed constraints and features.

VRP is one of the combinatorial optimization problems in the problem library provided by HyFlex [29], see Appendix A.1, with different types of perturbative *llh* (*mutation*, *ruin and recreate*, *local search* and *crossover*). An empirical study is conducted in the HyFlex framework [177] using an iterated local search selection HH with multi-armed bandit for VRPTW variants (Solomon, Gehring-Homberger, see Appendix B.3), as well as course timetabling problems. Statistical analysis and fitness-landscape-probing techniques are used on a set of training instances to identify a compact subset of the eight most effective *llh*. It is found that operators' evolvability (number of neighbours with better or equal fitness) can be used as an indicator to distinguish and select *llh* within HH. Using HyFlex, an iterated local search is employed in [191] as the high-level search to select from the 12 perturbative *llh* using online learning. In [8], offline apprenticeship learning is used in HyFlex to train classifiers based on small VRP instances to select from the 10 *llh* to improve solutions for unseen instances.

In [166], VRPTW problems are decomposed into sub-problems. Sub-solutions generated using column generation are combined and improved by a selection HH by selecting from seven perturbative *llh* for the benchmark Gehring-Homberger VRP (Appendix B.3. A multi-armed-bandit method is used based on online accumulative rewards, and solutions are accepted using a Monte Carlo mechanism.

7.3.2 Selection Hyper-Heuristics with Both Constructive and Perturbative Low-Level Heuristics

Some of the selection HH approaches employ meta-heuristics at the high level to select both constructive and perturbative *llh*. The high-level heuristics usually explore or configure sequences of *llh* operators, which are applied to construct and iteratively improve direct VRP solutions.

In [113], heuristic rules are used in a multi-agent system to select perturbative *llh* to improve solutions for distance-constrained VRP. Within an agent meta-heuristic framework, a coalition of agents explore the search space concurrently. In addition to learning by individual agents, the agents are also improved by exchanging information based on collective online learning. The *llh* operators are grouped as intensifier (improvement) or diversifier (generation, mutation and crossover) operators to strike a balance in the search. As the operators are used in both the initialization and during optimization, the HH can be seen as configuring and hybridizing both constructive and perturbative *llh*.

An evolutionary algorithm is developed in [66] at the high level to evolve sequences of constructive-perturbative *llh* pairs of variable lengths for 21 instances of Kilby's benchmark dynamic VRP [94]. Three types of *llh* are used, namely an ordering *llh* to rank customers, four constructive *llh* to construct solutions, and four repair heuristics to improve solutions. It is found that the *llh* evolved vary considerably for static and dynamic parts of the problem. It is crucial to design simple and effective *llh* considering adaption, average performance and speed. A diverse set of simple *llh* is recommended to improve the coordination among the *llh* during the evolution based on the communication of information on problem partial states.

In [122] a selection HH using an evolutionary algorithm is developed to evolve sequences of action units of *llh* (problem specific variations and mutations) like those in [66] for CVRP. It is observed that the best-evolved sequences of *llh* are composed of multiple actions of variations and mutations, indicating that larger neighbourhood structures are more effective at escaping local optima and producing high-quality solutions.

7.4 Generation Hyper-Heuristics for Vehicle Routing Problems

Compared to selection HH, generation HH aim to generate new heuristic functions or rules based on given *llh*. The newly generated functions or heuristics are then applied when solving new problems. The pre-defined set of *llh* usually includes problem attributes or features that are combined or configured using function operators or grammars by the high-level heuristics or methods. In the current literature, the most used high-level configuration methods for VRP are genetic programming [84, 175, 193, 104] and grammatical evolution [57, 164], using problem attributes (see Section 7.2.1) and genetic operators as shown in Table 7.1 for variants of benchmark VRP.

Genetic Programming HH In a two-level HH using GP, heuristic configurations at the high level are represented by trees, where terminal nodes representing problem attributes are connected by internal nodes representing function operators. The role of GP is to find the best heuristic configurations, usually by conducting training on a small subset of problem instances. In VRP, the design of GP consists of choosing a subset of the function set in Table 7.1, and the problem attributes as the terminal set (see Section 7.2.1). In [84, 193, 104] GP employs standard sub-tree crossover and mutation to evolve via generations new constructive heuristics based on selected training instances. The newly evolved heuristic functions or rules are applied to new instances of the same problem. Table 7.2 summarizes GP HH approaches for VRP.

Table 7.2 GP HH for VRP variants

	[193]	**[104]**	**[84]**	**[175]**
Terminals set	six attributes of vehicle behaviours	seven problem attributes	11 problem attributes	20 attributes and nine route selectors
Function set	$+, -, *, /, exp$ $max, sin, angle$	$+, -, *, /, min$ max	$+, -, *, /, max, exp$	$compare$
VRP variant	static and dynamic CARP	uncertain CARP	DVRP (Saint-Guillain in Appendix B.3)	CVRPTW (Solomon in Appendix B.3)
High-level *h*	Trees as heuristic functions h to calculate and assign the next vehicle (i.e. value for the next decision variable) until a complete solution s is constructed			
Low-level *s*	The direct solution, i.e. a set of routes, constructed using h			

In [193] the newly generated mathematical function is used to construct solutions of tours for five sets of capacitated arc routing problems (CARP), a counterpart of VRP where arcs rather than nodes are served by the vehicles. Six vehicle behaviours including demand, load, cost, satisfied, and depotCost, etc. (see Section 7.2.1) are used in the terminal set, configured and operated using eight mathematical operators (see Table 7.2, explained in Table 7.1) at the high level. The same problem objective function is used at both levels. The study also analyzed the features of the high-level search problem using two indirect representations in GP. The dimension of heuristic configurations is independent of problem size, and much smaller than that of direct solutions, as studied in [151]. The evolved new mathematical functions showed to perform well on both static and dynamic CARP.

A GP is developed based on training instances of Uncertain Capacitated Arc Routing Problems (UCVRP) with environment changes in [104]. The generated trees of new heuristics are used to order vehicle tasks during solution construction. Changes of task demand and edge accessibilities are addressed based on average and worst costs of the resulting solutions. Domain specific knowledge is studied to design effective GP HH. Three functions showed to be effective, by selecting a promising set of candidate tasks, detecting edge failures, and addressing route failures due to environment changes. Six function operators as shown in Table 7.2 are used, to configure problem attributes including demand, cost, load, depotCost, satisfied, and constant, etc.

Dynamic VRP with new arrival tasks is addressed in [84] using a GP HH. Average costs of training instances of nine different scenarios are used to measure the performance of HH to address the stochastic feature of real-time new task requests. In addition to a number of problem attributes including normalized travel time to the depot, normalized travel time from the current location, normalized service time, normalized demand, and vertex density, etc., two other terminals, the expected number of future requests and the probabilities of new requests, are also considered. The automatically generated heuristics are able to update routes with new arrival tasks, and significantly outperformed three manual heuristics. It is found that simple *llh* using probabilities as terminals does not improve the new heuristics generated.

In [175], GP is used to generate new constructive heuristics to construct solutions for a benchmark CVRPTW (Solomon in Appendix B.3) and a new real world VRP. A large number of problem attributes is considered in the terminal set. These can be catogerized into selecting nodes (average distance of the node to the remaining nodes, distance saving, first come fist served, slack in time window, time saving, and those in Section 7.2.1), and selecting routes (first route, least / most used route by time, route with most possible nodes, random route). Perturbative *llh*, local search and crossover operators are then selected by an iterated local search at the high level of HH to improve the initial solutions generated by the new heuristics. The newly generated heuristics alone showed to be competitive against seven standard constructive heuristics in VRP.

Grammatical Evolution HH In an HH using GE, high-level heuristic configurations are represented by a string genotype of variable length, i.e. a four-tuple Backus-Naur Form (BNF) $<T, N, S, P>$ (see Table 7.3). Starting with a symbol in S, a production rule P in the form of a BNF grammar sentence is composed based on user-defined terminals T connected by operators or methods in a non-terminal set N. P can be recursively interpreted until all elements in P are terminals in T, to generate executable programs for the VRP. The generated programs (parsed trees) are evolved by using genetic operators in GE.

Due to the nature of indirect genotype encoding in GE, the BNF grammar can be easily defined by users and the resulting trees are evolved using high-level configuration methods for solving different problems. The newly generated grammar thus potentially could be more easily reused for other problem instances compared to the newly generated constructive heuristics by genetic programming. In the literature, however, GE is less studied for VRP compared to genetic programming in generation HH; see Table 7.3.

Table 7.3 BNF grammars in GE HH for VRP

BNF	**Sabar et al. [164]**	**Drake, Killis and Özcan [57]**
Terminals set *T*	Neighbourhood operators and 11 symbols (*AllMoves* and *Great Deluge*, etc.)	26 attributes in Section 7.2.1
Non-terminal set *N*	Acceptance criteria, LST configurations, Neighbourhood structures / combinations	12 arithmetic and relational operators in Table 7.1
Start symbol *S*	<*LST*>: Local Search Template	<*Initialization*><*Ruin*><*Recreate*>
Production *P*	Rules composed of *S* followed by elements in *N* and *T*	Rules / sentences consists of elements in *N* and *T*
Problem	DVRP (Christofides and Golden [72], exam timetabling (Carter, ITC2007 in Appendix B.4)	CVRP, VRPTW (Augerat, Solomon in Appendix B.3)

In [57], heuristics generated by GE are used not only to build the initial solutions for a variable neighborhood search (VNS) algorithm, but also to select *ruin and insertion* operators within the VNS for two VRP variants. Instead of systematically switching operators in a predefined order in VNS, GE is used to automatically select operators during the search. Another GE HH is devised in [164], using perturbative *llh* to automatically generate and evolve templates of neighbourhood operators and eight acceptance criteria for both dynamic VRP and exam timetabling problems. The GE is associated with an adaptive memory of high-quality solutions to increase diversity, leading to high performance in both problem domains.

7.5 Discussion

Current research in HH for VRP has addressed a diverse range of interesting issues. Both perturbative and constructive *llh* have been hybridised and selected in one selection HH, providing general methods applicable to real-world VRP with various features. In selection HH, studies on identifying a compact subset of effective *llh* revealed ideas to build more efficient HH frameworks. Combined *llh* (larger neighbourhood operators) in selection perturbative HH showed to be more effective at escaping from local optima. Online and offline learning for dynamic VRP obtained more insights on dealing with uncertainties in VRP, and increased the robustness and generality of HH. HH approaches serve as the mechanisms to explore and configure integrated approaches based on either online or offline learning, indicating promising future developments across both HH and meta-heuristics.

On the other hand, although generation constructive HH has been successfully applied to generate new heuristics or functions for variants of VRP, more research needs to be conducted to obtain in-depth analysis on generating new effective constructive and perturbative heuristics for different instances or problem variants.

Compared to other less-studied problem domains such as nurse rostering, VRP variants have attracted relatively more attention in HH partially due to the wide range of different constraints and problem features. Due to the high demand for its real-world applications, however, there is much more potential in HH to provide general solution methods for variants of VRP.

In VRP, perturbative *llh* configured by different selection HH approaches (evolutionary algorithms, iterated local search, and classifiers, etc.) have showed to be applicable directly to different VRP instances or variants. Automatically generated new constructive heuristics configured using offline learning on training instances based on problem state features, however, raise the question of reusability to variants of VRP. More analysis is needed on how the same generation HH framework with different constructive *llh* could be reused, with offline or online learning, to provide more insight to address this issue.

More findings of *llh* can facilitate more general and effective HH for a wider range of VRP applications. This chapter focuses on benchmark VRP with several different constraints or features in VRP variants. The high-level search in HH configures *llh*, which encapsulate details in the specific problem. A range of *llh* for VRP have been investigated, either on problem attributes with function operators in genetic programming or grammatical evolution, general move operators in local search, or genetic operators in evolutionary algorithms. Based on emerged findings, some of these *llh* can be extended to encapsulate other problem specific features, constraints and uncertainties, and applied to real VRP. A library of simple elementary *llh*, categorized to address specific aspects of vehicle / customer behaviour and constraints, focus of search (intensification or diversification), and acceptance criteria, and easily portable and extendable (i.e. via the form of XML) can facilitate designing such more effective and general HH frameworks for real-world VRP.