A study on memetic computation, with applications to capacitated vehicle routing problems

Chen, Xianshun

2014

Chen, X. (2014). A study on memetic computation, with applications to capacitated vehicle routing problems. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/55386>

<https://doi.org/10.32657/10356/55386>

NANYANG TECHNOLOGICAL UNIVERSITY

**A STUDY ON MEMETIC
COMPUTATION, WITH
APPLICATIONS TO CAPACITATED
VEHICLE ROUTING PROBLEMS**

CHEN XIANSHUN

School of Computer Engineering

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

2014

Abstract

Memetic computation, a paradigm that *uses the notion of meme(s) as units of information encoded in computational representations for the purpose of problem-solving*, represents one of the frontiers in the computational intelligence research, of today's optimization research. This dissertation takes an explorative attitude in the design of memetic computing frameworks and algorithms by leveraging the co-adaptive nature of meme complexes or memeplexes. The current research presented in this dissertation takes multi-memes and memeplexes as focal point of interest in the context of memetic computation and develop meme-centric computing frameworks for more effective problem-solving in the context of stochastic optimization of *Capacitated Vehicle Routing Problems (CVRP)* and *Vehicle Routing Problem with Stochastic Demands (VRPSD)*.

The presented research works follow a natural progression from multi-memes to memeplexes in simple and adaptive hybrids. Firstly, a simple hybrid with a multi-memes individual learning was developed to overcome the inherent risk associated with reliance on a single meme to be used for individual learning in memetic computation for solving CVRP. The proposed simple hybrid invests local search time budget among multiple memes with different search biases during individual learning, in order to reduce risks of failing on problems from different classes as well as make finding high quality solutions more likely. On this basis, an adaptive hybrid with the conceptual modeling of memeplexes was explored as a self-configuring methodology capable of learning and adapting multi-memes individual learning to the given problem of interests while the search progresses online. The connection topology network of the memeplex representation, credit assignment criteria for evaluating individual memes and meme co-adaptation, as well as the role of emergent memeplexes in the individual learning process were explored, with its application for solving CVRPs of diverse characteristics. Finally, a memeplex robust solution scheme which seeks to achieve greater level of adaptivity in

problems with uncertainty was proposed and studied for VRPSD. Via robust solution search scheme, memplex individual learning and gene-meme coevolutionary model, the scheme was demonstrated to be highly efficient and effective problem solver in environments with uncertainty such as VRPSD.

Acknowledgments

I would like to express my deepest gratitude to my academic supervisor in Nanyang Technological University (NTU), **Associate Professor Ong Yew Soon**.

His sincere encouragement during my indecisive moments and inspirational insights into the many aspects of my research direction have faithfully guided me throughout my research work. Without Associate Professor Ong's guidance, the fruitful achievements of this project would not have been possible.

I would also like to thank Associate Professor Lim Meng Hiot for his beneficial comments and support.

In addition, I would like to thank my friends for their encouragements and the staffs in Intelligent Systems Centre (IntelliSys) and Center for Computational Intelligence for their technical supports.

Last but not least, special thanks to my parents for their constant support, understanding and love.

Chen Xianshun

Contents

Abstract	i
Acknowledgments	iii
List of Abbreviation	vii
List of Symbols	viii
List of Figures	x
List of Tables	xii
1 Introduction	1
1.1 Research Objective	1
1.2 Organization of the Thesis	2
2 A Multi-Facet Survey on Memetic Computation	4
2.1 Simple Hybrids	8
2.1.1 Types of Search Methods	9
2.1.2 Level of Hybridization	11
2.1.3 Modes of Inheritance	12
2.1.4 Types of Domain Knowledge Incorporated	14
2.2 Adaptive Hybrids	16
2.2.1 Diversity Management	16
2.2.2 Design Adaptation Issues	18
2.2.3 Theoretical Analysis of MA	20
2.3 Concluding Remarks	21

3	Multi-Memes Individual Learning in Simple Hybrids	23
3.1	Introduction	23
3.2	Multi-Memes Individual Learning	25
3.3	Capacitated Vehicle Routing Problem	30
3.3.1	Problem Formulation	31
3.3.2	Literature Review	34
3.4	Cooperating Memes Memetic Algorithm for solving CVRP	35
3.4.1	Population-based Evolutionary Search	36
3.4.2	Multi-Memes Individual Learning for CVRP	39
3.5	Empirical Study	45
3.6	Conclusions	56
4	A Conceptual Modeling of Meme Complexes in Adaptive Hybrids	57
4.1	Introduction	57
4.2	Memplex in Memetic Computation	61
4.2.1	Memplex Representation	62
4.2.2	Meme Co-adaptation	65
4.3	Role of Memplex in Individual Learning	67
4.3.1	Credit Assignment for Individual Memes	71
4.3.2	Credit Assignment for Co-adapted Memes	72
4.3.3	Emergence of Memplexes	75
4.4	Co-Adapted Memetic Algorithm for solving CVRP Problems	78
4.4.1	Memplex Representation for Vehicle Routing Problem	80
4.5	Empirical Study	81
4.5.1	Search Quality and Effectiveness of CAMA	83
4.5.2	Computational Cost and Efficiency	92
4.5.3	CAMA against other recent state-of-the-art meta-heuristics algorithms in the literature	94
4.6	Conclusions	101

5	A Memplex Robust Search Scheme	103
5.1	Introduction	103
5.2	Vehicle Routing Problem with Stochastic Demands	104
5.2.1	Problem Formulation for VRPSD	106
5.2.2	Safety Stock for Managerial Control over Route Failure in VRPSD . . .	107
5.3	Memplex Robust Search Methodology	108
5.3.1	Genetic and Memetic Representations	109
5.3.2	Fitness Evaluation based on Robust Solution Search Scheme	113
5.3.3	Individual Learning using Memplex	115
5.3.4	Gene-Meme coevolution	118
5.3.5	Genetic Representation Repair	121
5.3.6	Workflow of SAMRS	121
5.4	Empirical Study	124
5.4.1	Experimental Setup	124
5.4.2	Results and Discussion	125
5.5	Conclusion	132
6	Concluding Remarks	134
6.1	Research Contributions	134
6.2	Future Work	136
6.2.1	Multi-objective/Constrained Optimization	136
6.2.2	Optimization of Dynamic Problems	137
6.2.3	Continuous Optimization	137
6.2.4	Theoretical Study on Memplex Emergence	138
6.2.5	Towards Memetic Automaton	138
	References	140

List of Related Publications

List of Abbreviations

VRP	Vehicle Routing Problem
CVRP	Capacitated Vehicle Routing Problem
CAMA	Co-Adaptive <i>Memetic</i> Algorithm
CMMA	Cooperating <i>Memes Memetic</i> Algorithm
SAMRS	Stochastic-Adaptive <i>Memplex Robust Search</i>
EA	Evolutionary Algorithm
ACO	Ant Colony Optimization
PSO	Particle Swarm Optimization
EP	Evolutionary Programming
ES	Evolution Strategy
GA	Genetic Algorithm
GP	Genetic Programming
MA	Memetic Algorithm

List of Symbols

$\mathcal{E}_{\mathcal{N}}$	The set of N customers $\mathcal{E}_{\mathcal{N}}$ in a VRP
$\mathcal{V}_{\mathcal{K}}$	The fleet of K vehicles in a VRP
e_l	A customer with index l belonging to $\mathcal{E}_{\mathcal{N}}$
v_k	A vehicle with index k belonging to $\mathcal{V}_{\mathcal{K}}$
τ_k	The route traveled by the vehicle v_k
$\hat{\tau}_k$	A sub-route of the route τ_k
a_{kj}	The j^{th} customer visited by v_k in the route τ_k
C_v	The limited capacity of a vehicle in a VRP
L_{max}	The maximum route length a vehicle can travel in a VRP
s	A solution in a VRP
s_{opt}	An optimal solution in a VRP
g	The average number of evolutionary search generations elapsed before the solution converged
$\text{Pop}(g)$	The current population of solutions at generation g in a memetic algorithm
M	The size of population $P(g)$
s_i	A solution width index i in the population $P(t)$
$\mathcal{M}_{\pi}$	A memplex candidate having a memplex policy π
$\mathcal{M}_i$	A memplex associated with solution s_i
$L(\tau_k)$	The length of route τ_k , which is the distance traveled by v_k and $a_{k(j+1)}$
$d(a_{kj}, a_{k(j+1)})$	the distance between two consecutive customers a_{kj} visited by v_k
$C(s)$	The cost of solution s (which in a VRP, represents overall distance traveled by all vehicles as specified in solution s)
$D(\tau_k)$	The total demand on route τ_k , as serviced by vehicle v_k

$\Delta C = C(s) - C(s')$	Reduction in cost when solution s is transformed to s'
Ψ	The terminal set of memes $\{m_k\}_k$
T_i	A move operator defined for refining a solution
C	The best fitness among the converged solutions by an algorithm on a problem instance
$\bar{C}$	The average fitness of the converged solutions obtained by an algorithm on a problem instance
C_{best}	The best known solution fitness of a problem instance as reported in the literature
gap	The minimum percentage fitness deviation from C_{best} as given by $\frac{C - C_{best}}{C_{best}} \times 100\%$
$\overline{gap}$	The average percentage fitness deviation from C_{best} as given by $\frac{\bar{C} - C_{best}}{C_{best}} \times 100\%$
t	The computation time, in seconds, to arrive at the converged solution
$\bar{t}$	The average computation time, in seconds, to arrive at the converged solutions
p_t	p -value denotes the <i>two-tailed</i> 5% t-test solution fitness difference between two algorithms
p_r	p -value denotes the <i>right-tailed</i> 5% t-test solution difference between two algorithms
k	The safety stock level in a Vehicle Routing Problem with Stochastic Demands

List of Figures

2.1	Levels of Hybridization in MA	11
3.1	Workflows of Single-meme Individual-Learning versus Multi-memes Individual-Learning	29
3.2	Multi-Route VRP Solution Representation	32
3.3	Genetic Crossover between s_i and s_j (phenotype)	37
3.4	Move Operators: T_1, T_2, T_3	41
3.5	Move Operators: T_4, T_5, T_6	42
3.6	Plot of minimum and average percentage fitness deviations from the best known results across Augerat benchmarks	52
3.7	Plot of minimum and average percentage fitness deviations from the best known results across CE benchmarks	53
3.8	Plot of minimum percentage fitness deviation from the best known results across Golden benchmarks	54
3.9	Plot of minimum percentage fitness deviation from the best known results across Christofides benchmarks	55
4.1	Memeplex in its neural wiring structure	63
4.2	The network topology of memeplex candidate displaying part of connectivities	65
4.3	Workflow Comparison of Individual-Learning by Meme Adaptation and Memeplex	69
4.4	The credit assignment for co-adapted memes as $\mathcal{M}_\pi$ refines a solution	74
4.5	Minimum and Average percentage fitness gaps, $gap, \overline{gap}$, across Augerat & CE benchmarks (empty bars implies zero gap readings)	85

4.6	Minimum and Average percentage fitness gaps, $gap, \overline{gap}$, across Christofides benchmarks (empty bars implies zero gap readings)	86
4.7	Minimum and Average percentage fitness gaps, $gap, \overline{gap}$, across Golden benchmarks (empty bars implies zero gap readings)	87
4.8	Plot of average time required for convergence across Augerat and CE benchmarks	93
4.9	Plot of average time required for convergence across Christofides benchmarks . . .	93
4.10	Plot of average time required for convergence across Golden benchmarks . . .	94
4.11	Minimum and Average percentage fitness gaps, $gap, \overline{gap}$, across Augerat and CE benchmarks	99
4.12	Minimum percentage fitness gaps, gap , across Golden benchmarks	100
4.13	Minimum percentage fitness gaps, gap , across Christofides benchmarks (c1 and c12 omitted due to their insignificant performance)	101
5.1	Outlines of SAMRS and its components	109
5.2	Genetic and Memetic Representations of $agt_i = \{s_i, \mathcal{M}_i\}$: s_i with the decoded routes, safety stock level k_i and $\mathcal{M}_i$ with search instructions	112
5.3	Gene-Meme coevolution Cycle among $s_i, \mathcal{M}_i$ in $\text{Pop}(g)$	119
5.4	Memetic Recombination between $\mathcal{M}_i$ and $\mathcal{M}_j$ (top: memotype bottom: socio-type)	120
5.5	Gaps for solution at different variance levels (CVRP-best $\equiv Gap_0$; OBS-best $\equiv Gap_1$; SAMRS-best $\equiv Gap_2$; SAMRS-avg $\equiv \overline{Gap_2}$)	129
5.6	Reliability indices for solution at different variance levels (CVRP-best $\equiv R_0$; OBS-best $\equiv R_1$; SAMRS-best $\equiv R_2$; SAMRS-avg $\equiv \overline{R_2}$)	130

List of Tables

3.1	Results on Augerat benchmarks (The best performing algorithm of the respective CVRP instance is highlighted in Bold)	47
3.2	Results on CE benchmarks (The best performing algorithm of the respective CVRP instance is highlighted in Bold)	47
3.3	Results on Christofides benchmarks (The best performing algorithm of the respective CVRP instance is highlighted in Bold)	48
3.4	Results on Golden benchmarks [Pb1-Pb10] (The best performing algorithm of the respective CVRP instance is highlighted in Bold)	49
3.5	Results on Golden benchmarks [Pb11-Pb20] (The best performing algorithm of the respective CVRP instance is highlighted in Bold)	50
4.1	Comparison among CAMA, SGMA, MultiMeme on Augerat Benchmarks ⁽⁺⁾ indicates <i>CAMA</i> performs better statistically, in terms of p_t and p_r respectively)	84
4.2	Comparison among CAMA, SGMA, MultiMeme on CE Benchmarks, ⁽⁺⁾ indicates <i>CAMA</i> performs better statistically, in terms of p_t and p_r	84
4.3	Comparison among CAMA, SGMA, MultiMeme on Christofides Benchmarks, ⁽⁺⁾ indicates <i>CAMA</i> performs better statistically, in terms of p_t and p_r	88
4.4	Comparison among CAMA, SGMA, MultiMeme on Golden Benchmarks, ⁽⁺⁾ indicates <i>CAMA</i> performs better statistically, in terms of p_t and p_r	89
4.5	Success Rate of CAMA, SGMA, MultiMeme ($\checkmark \equiv 100\%$)	91
4.6	Converged Solution Quality of the Algorithms on the Augerat and CE benchmarks, the best performing algorithm of the respective CVRP instance is highlighted in Bold	96
4.7	Converged Solution Quality of the Algorithms on Christofides benchmarks, the best performing algorithm of the respective CVRP instance is highlighted in Bold	97

4.8	Converged Solution Quality of the Algorithms on Golden benchmarks, the best performing algorithm of the respective CVRP instance is highlighted in Bold	98
5.1	Results of the 55 VRPSD instances in a low-variance scenario $\alpha = 0.05$ (The best performing algorithm of the respective VRPSD instance is highlighted in Bold)	126
5.2	Results of the 55 VRPSD instances in a medium-variance scenario $\alpha = 0.25$ (The best performing algorithm of the respective VRPSD instance is highlighted in Bold)	127
5.3	Results of the 55 VRPSD instances in a high-variance scenario $\alpha = 0.75$ (The best performing algorithm of the respective VRPSD instance is highlighted in Bold)	128

Chapter 1

Introduction

Optimization theory [65] is an interdisciplinary branch of applied mathematics and formal science that uses methods such as mathematical modeling, statistics, and algorithms to arrive at optimal or near optimal solutions to complex problems. One of the frontiers in today's optimization research is memetic computation. Memetic computation is *a paradigm that uses the notion of meme(s) as units of information, i.e., the building blocks, encoded in computational representations for the purpose of problem-solving* [148, 28].

Memetic computation covers a plethora of potentially rich meme-inspired computing methodologies, which have been demonstrated to converge to high quality solutions more efficiently than their conventional counterparts on a wide range of real world problems.

1.1 Research Objective

Despite the many reported successes on significant instantiations of memetic computation across a wide range of application domains, from the complexity point of view, the diversity of memetic representational space that has been explored to date for the construction of notable and flexible memetic hybrids has remained to be surprisingly small. In other words, the potentials of the “meme” metaphor, which have inspired the developments of memetic computation, have not been fully developed to their full potential for the purpose of problem solving.

In this research work, novel and flexible memetic representations of *multi-memes* and *memeplexes*¹, have been explored for the purpose of crafting more effective problem-solving methodologies in the context of memetic computation. The objective of this research is thus to establish a *meme-centric* framework that takes *memeplexes* as the focal point of interest in memetic computation, through a seamless integration of memes as units of domain information useful for problem-solving, so as to attain a higher level of memetic representations addressing the ever increasing complexity and dynamic nature of problem-solving.

Working with *Capacitated Vehicle Routing Problem* (CVRP) and its more complex *Vehicle Routing Problem with Stochastic Demand* (VRPSD) variant, this research investigates on various forms of memetic computational frameworks such as simple hybrids and adaptive hybrids (refers to Chapter 2). By capitalizing on the meme co-adaptive nature of memeplexes, memetic methodologies are proposed and demonstrated to converge more efficiently at high quality solutions than existing state-of-the-art algorithms for solving CVRP and VRPSD.

1.2 Organization of the Thesis

The remainder of this thesis is organized as follows:

- In Chapter 2, a comprehensive multi-facet survey of recent research developments in memetic computation is presented, covering a plethora of memetic hybrids. This chapter serves to facilitate the background knowledge for ease of understanding.
- Chapter 3 presents a study to enhance the problem-solving capability of simple hybrid with a multi-memes individual learning, combining multiple distinct memes of unique search biases in a synergistic manner.

¹In science, a memeplex represents an ideology or belief system, e.g. religion, that exist in the mental state of an individual during its lifetime. Structurally, a memeplex is a co-adapted stable set of mutually assisting memes [40], and can arise in the meme pool of seemingly unconnected memes [16].

- Chapter 4 presents a study on adaptive hybrid with the conceptual modeling of memeplex individual learning, as an extension of the multi-memes individual learning in Chapter 3 capable of learning and adapting to the given problem of interests such as CVRP, while the search progresses online.
- Chapter 5 presents a study on the memeplex robust solution scheme which seeks to achieve a greater level of adaptivity in problem environments with uncertainty such as VRPSD, via robust solution search scheme, memeplex individual learning and gene-meme coevolutionary model.
- Finally, Chapter 6 concludes this thesis and outlines potential future research directions in the area of memetic computation that could leverage from the meme co-adaptive nature of memeplexes as well as the applications to capacitated vehicle routing problems.

Chapter 2

A Multi-Facet Survey on Memetic Computation

For the past three decades, many population based search techniques have surfaced to become a mainstay of optimization [203, 235, 84, 124]. These techniques rely on search ideologies that work on the basis of manipulating samples representative of the search sub-regions within the solution landscape. Such inherently parallel or multi-track search algorithms derive their strength from the simultaneous explorations of multiple search regions and exploitative interactions or coupling between the different search tracks.

Typically, each search track manifests itself as an individual within a population which represents a potential solution to the problem that the algorithm is intended to solve [235, 163, 46, 83]. The transitional states of each individual are governed by nature-inspired processes such as evolutionary and swarm intelligence including but not limited to Genetic Algorithm [126], Evolution Strategy [14], Evolutionary Programming [52], Genetic Programming [166], Differential Evolution [169], Estimation of Distribution Algorithm [107], Ant Colony Optimization [43], Particle Swarm Optimization [165], and Artificial Immune System [39], which are generally classified under two main factors; modes of replacement for each individual and the guiding principles dictating the rule of behavior or interactions of the population as a whole [149, 157, 98].

Individuals within a population go through processes that operate on them. These processes are usually effected through operations that manipulate the solution coding in the form of strings (binary bit string, integer permutation, etc.), trees or graphs; most notably with selection (such as ranking, roulette wheel, stochastic sampling, tournament, etc.) and reproduction (crossover, mutation, elitism, etc.). Among the three main operations, mutation serves to introduce random variations into the population, essentially a mechanism to circumvent population stagnation. What is evident is that such processes introduce well-defined procedural manipulations of the solution.

These days the evolutionary processes are typically inspired metaphorically by processes observed in nature, more specifically in population genetics. One should take care to note that such processes usually do not instill any higher level domain or problem-specific information beyond that required to evaluate the fitness of the individuals in the population. Recent trends in research on optimization have biased towards algorithms that incorporate higher level of evolutionary and adaptive behavior [149, 157, 98, 24, 21]. As with genetics, the science of memetics [16] has served as a motivational pillar and inspiration towards the possibility of *meme* developing into a proper hypothesis of the human mind. In the context of computing [148], memes denote the recurring real-world patterns or domain specific knowledge encoded in computational representations for the purpose of effective problem-solving. In the last decade, there are signs of increasing effort in research on evolutionary methodologies inspired by population memetics that enhance the capacity of algorithms in establishing a good balance of explorative and exploitative traits.

As with genes in genetics, a meme is synonymous to memetics as being a building block of cultural know-how that is transmissible and replicable. While genes form the “*instructions for building proteins*”, memes are “*instructions for carrying out behavior, stored in brains (or other objects)*” and passed on by means such as imitation [16]. The term *meme* can be traced back to Dawkins [40] in his book “The Selfish Gene”. The term has inspired the new

science of memetics which today represents the mind-universe analog to genetics in cultural evolution, stretching across the fields of anthropology, biology, cognition, psychology, sociology and socio-biology [16]. In computer science and engineering, the meme-inspired computing methodology or more concisely *memetic computation* has become an increasing focus of research in recent years [148]. One of the most direct applications of memetic computation in problem-solving, in recent years, has been *memetic algorithms* [155]. It is notably one of the simplest forms of memetics inspired techniques, establishing itself as one of the key methodologies in complex optimization and operational research tackling large-scale, real-world search problems.

The term *memetic algorithms* (MAs) originated from Moscato [133] as being the algorithmic pairing of a population-based search method with one or more refinement methods. MAs have been successfully applied to a wide range of domains that cover problems in combinatorial optimization [78, 207, 176], continuous optimization [150, 147], dynamic optimization [62, 84], and multi-objective optimization [93, 203, 61]. In particular, remarkable success on significant instantiations of MAs across a wide range of real-world applications has been reported, ranging from the field of bioinformatics [178, 54], permutation flow shop scheduling [78], scheduling and routing [207, 176], to nonlinear programming problems including aerodynamic design [150], atomic and molecular structure problems [140], optimal control systems with machine learning [77, 131, 23], and computationally expensive and uncertain environments [152, 82].

It is clear that MA has managed to stir up a wide interest in the field of evolutionary optimization algorithms [154] and at times, it is an easy source of confusion whereby any meme-inspired paradigms are conveniently taken to be synonymous to MA [148]. There are potentially many other rich meme-inspired designs, operational models, and algorithm frameworks which form the cornerstones of memetic computation as tools for problem-solving [116, 148, 84, 33, 157, 218, 47, 48]. To capture the essence of existing and potential research

in this field, it is necessary to clarify memetic computation as being *a paradigm that uses the notion of meme(s) as units of information, i.e., the building blocks, encoded in computational representations for the purpose of problem-solving*. From the algorithmic point of view, memes are units of domain specific information useful for problem-solving. For instance, in hybrid algorithms, memes are taken as instructions, rules, strategies, *a priori* knowledge, etc., for search. In a more general problem-solving context, memes are encoded in various computational representations of the problem solvers which acquire increasing level of problem-solving capability.

To date, work on memetic computation can be divided into several unique categories for the sake of brevity. In this chapter, these works are classified from the perspective of simple hybrids and adaptive hybrids. Simple hybrids like MAs [133] represent a form of synergistic combination between population-based and local refinement heuristics. The rationale is that, often the performance of population-based search such as evolutionary algorithms can be enhanced if problem-specific refinement techniques are incorporated. These simple hybrids incorporate domain-specific information to augment the population-based with refinement components. The term memetic algorithm has stemmed from the fact that the local refinement procedure is akin to a meme which represents some form of domain-specific *a priori* knowledge of the human expert on how the solution can be better refined [133]. Since the individual in the population is perceived as undergoing a form of continuous refinement, the process is often also referred to as local search, lifetime learning or individual learning.

Although remarkable success on significant instantiations of specialized MAs across a wide range of application domains, researchers have also actively ventured into the emerging field of adaptive memetic algorithms. Adaptive hybrids [149, 157, 98, 221, 24, 21] are a class of memetic computation methodologies with great capacity of acclimatizing to suit a given problem in hand, by methodically utilizing acquired information about the matching of problems to procedures, and reconfiguring itself to adapt to the domain or instances of problems being

solved. The potential algorithmic improvements have been attempted by adapting several key design issues including frequency of refinement, selection of the individual subset for refinement, intensity of refinement, choice of meme to undergo refinement as well as the choice of population-based search and fitness functions [136].

The purpose of this chapter is to present a comprehensive multi-facet exposition of recent research in memetic computation. For the sake of completeness, the chapter begins in sections 2.1 and 2.2 with a brief review on the early manifestations of memetic computation within the context of evolutionary computation, often construed as memetic algorithms, hybrids and adaptive hybrids. To give a multi-facet survey of memetic computation, a review on the more recent memetic algorithms, hybrids and adaptive hybrids is provided in the chapter. In particular, the current review takes focus on the recent developments of hybrids and adaptive hybrids, during which there is a clear exponential growth of research publications in the area. Lastly, we conclude this chapter with some remarks on memetic computation.

2.1 Simple Hybrids

Early works in memetic computation have materialized in the form of hybridization between the population-based search and refinement procedures [99]. They are also often known as genetic local search, canonical memetic algorithm or simply as hybrids. From the algorithmic perspective, two or more distinct methods when combined together in a synergistic manner with the incorporation of domain knowledge [153], can greatly enhance the problem-solving capability of the derived hybrid. Furthermore, hybrids capitalize on the complementary advantage of population-based search (more explorative) and refinement (more exploitative) in that the former provides a reliable estimate of the global optimum while the latter concentrates the search effort around the best solutions found so far by searching their neighborhoods to produce better solutions more efficiently [149]. As such, hybridization is one important feature evident in the field of memetic computation. In this section, the main issues in simple hybrids

are discussed, including the types of population-based search methods and refinement procedures, the levels of hybridization, modes of inheritance as well as the types of domain-specific information incorporated into simple hybrids.

2.1.1 Types of Search Methods

Here some of the well-established population-based search approaches typically used in the literature, and the refinement approaches used to enhance them, are discussed in the form of hybrids. What constitutes the population - based search and refinement procedure in simple hybrids is motivated by the presence of *a priori* knowledge on the problem domain.

Typical population-based search frameworks including Genetic Algorithm (GA) [126], Evolution Strategy (ES) [14], Evolutionary Programming [52], Genetic Programming [166], Differential Evolution (DE) [169], Estimation of Distribution Algorithm (EDA) [107], Ant Colony Optimization (ACO) [43], Particle Swarm Optimization (PSO) [165], Artificial Immune System [39], etc., are stochastic in nature [235]. These are generic population-based search frameworks applicable to many problem domains. However, due to the ease of use, some are observed to be favored in particular domains. ACO and GA have stronger presence in the domain of order-based discrete combinatorial optimization (optimization problems which find optimal solution from a discrete set of feasible solutions, each of which may be represented by a permutation string, e.g., TSP and VRP), while DE, EDA, PSO are more often used for handling continuous variable problems.

In the domain of multi-objective optimization (MOO) [83, 33, 93], the more popular population-based methods include the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [41], Evolutionary Multiobjective Optimization (EMO) algorithms [78], Pareto Archived Evolutionary Strategy (PAES) [92], Multi-objective PSO [117], etc. It is worth noting that to date many of the successful multi-objective evolutionary algorithms (MOEA) involve some forms of hybridizations that are specially designed for dealing with their respective challenges of interest

[61, 78, 93]. Special-purpose reproduction operators [76], specialized fitness function management schemes [237, 196, 228], multi-objective refinement operators [78, 73], etc., are among some of those introduced into multi-objective MA (MOMA) hybrids, as a way of improving, the rate of convergence to the Pareto optimum solutions, spread of solutions along the Pareto front, or dealing with the presence of non-linear constraints.

The refinement procedure used in hybrids on the other hand, come in the form of both deterministic [51] and stochastic [197]. Among the deterministic refinement approaches, branch and bound has been used for handling combinatorial problems such as longest common subsequence [18], while Hooke-Jeeves algorithm, Nelder-Mead simplex search method, Rosenbrock algorithm, etc., for handling continuous problems such as complex engineering design [25]. To cope with the scale-up in problem complexity, such as high multi-modality, researchers have ventured into stochastic refinement to enhancing search diversity in the neighborhood. In combinatorial optimization domain, some of these include the tabu search for finding low auto-correlation binary sequences [53], simulated annealing to discover the optimal resources in peer-to-peer networks [134], etc. While many of these refinement methods have been designed for single-objective optimization, others specifically for multi-objective optimization have also emerged [75, 204, 92, 83]. One of the earlier works in MOMA [75] proposed a local search operator that perturbs the solutions in a randomly chosen non-dominated improving direction of the objectives, as a means of enhancing the rate of convergence to the Pareto front. Refinement procedure that directs the search towards and along the (local) Pareto set with the aim of obtaining a wider spread of solutions in the Pareto front like the Hill Climber with Sidestep [106] is among some of the others introduced recently. Other specialized refinement procedures proposed for mitigating the impact of uncertain search environments or noisy problems include the trust-region derivative-free memetic algorithms [210, 151, 68, 238].

2.1.2 Level of Hybridization

In this section the different refinement levels of hybridization used in MA are discussed. Researchers have incorporated refinement procedures at different stages of the population-based search. Most notably, it is possible to classify refinements as being incorporated *before*, *after*, or *interleaved* as depicted in Fig. 2.1.

Figure 2.1: Levels of Hybridization in MA

Refinement incorporated *before* the population-based search in the form of initialization has been shown to enhance the efficiency of simple hybrids. S. Rahnamayan [171] proposed the opposition-based learning to initialize the foremost population, and showed that in contrast to random initialization, their proposed population initialization method helped to accelerate the search convergence rate. In the same spirit, [88] demonstrated search improvements with a sequential transformation method for population initialization in the context of travelling salesman problem (TSP), while [132] with a scheme based on population dispersion to effect

the initialization. On the other hand, for procedures that are interleaved within the population-based search operators, refinement is conducted only after one or more reproduction operators have completed [73]. Note that the refinement process can be used more than once within the population-based search [66]. For other hybrids that incorporate the refinement process as a form of post-processing, [91] used inductive logic programming for improving the solutions in capacitated location routing problem and Steiner tree problem, respectively. The use of population-based search to coordinate interleaving refinement procedures [157] or fine-tuning the configurations of refinement procedures [239] to suit an optimization problem in hand, have also been considered. These include the hyper-heuristics [157], meta-Lamarckian MA [147], Indirect EAs [239], where the latter manages a collection of low-level heuristics as refinement procedures.

As illustrated in Fig. 2.1, the hybrids can also be deployed in the form of multi-populations or multi-islands or ensembles [193, 123, 161, 233]. These independent hybrid models cooperate through the exchange of genetic or memetic information or otherwise. The improved robustness, diversity and efficiency of multi-population hybrids have been reported by many [161, 233].

2.1.3 Modes of Inheritance

In this section, the modes of inheritance used in hybrids are reviewed. Two modes of inheritance have been studied closely in hybrids, namely *Lamarckian* and *Baldwinian* inheritances. Lamarckian inheritance [147, 47] characterizes that an individual passes on the learned traits acquired in its lifetime to its offspring directly. On the other hand in Baldwinian inheritance [9, 118], the learned traits as acquired are not passed to the offspring. Instead, learned traits result in the increased fitness of individuals, thus giving fitter learned individuals higher chance of survival than naive Darwinian evolution.

In the context of computational intelligence, Lamarckian learning forces the genotype to reflect the result of improvement through placing the refined and improved individual back into

the population to compete for reproductive opportunities. Baldwinian inheritance, on the other hand, only reflects the refinement improvement of individuals in fitness change while leaving the genotype of the population unaffected. Let $x, x^{opt} \in X^N$ where x and x^{opt} denote initial solution vector and improved solution vector upon undergoing refinement, X^N is the solution space of dimension N with fitness function $f(\cdot)$. Algorithmically, Lamarckian and Baldwinian learning effects the following changes $Lamarckian(x, f(x)) \rightarrow (x^{opt}, f(x^{opt})), \{x \in X^N\}$ and $Baldwinian(x, f(x)) \rightarrow (x, f(x^{opt})), \{x \in X^N\}$, respectively. The majority of simple hybrids in memetic computation usually employ the mode of Lamarckian inheritance [147, 47] in their design.

While Lamarckian learning is known to be more effective than Baldwinian learning under static environments [184], it has the known effect of disrupting schema processing since it changes the genetic structure of individuals, thus prone to early premature convergence [226]. Baldwinian, on the other hand, tends to induce a slower schema disruption than Lamarckian inheritance, and is known for its hindering effect that tends to block out any genetic differences [45]. In practice, both modes have found their places in simple hybrids where Lamarckian inheritance exhibited excellent advantage on unimodal landscape and problems with non-changing environments, while Baldwinian inheritance is deemed as more appropriate for problems subjected to dynamic and uncertain environments [183, 12, 152].

On real-world optimization problems subjected to noise or uncertainties in the design variables and environmental conditions, or time-varying fitness functions [183], such as *robust* aerodynamic shape design [152] and dynamic neural network evolution [12], hybrids that employed Baldwinian inheritance have reported notable search performances. Simulation studies such as [160] have also been made by some researchers to verify the higher robustness of Baldwinian inheritance than Lamarckian inheritance under dynamic environment. Further comparison between the two modes of inheritance was also reported in [74], whereby Lamarckian and Baldwinian were served as repair schemes for computational experiments on multi-objective

0/1 knapsack problems. Results obtained indicated that Baldwinian outperformed Lamarckian repair. Mixes of Baldwinian and Lamarckian inheritances in hybrids have also been investigated. One such instance in [29] reported improved efficiency in the search involving sorting network problem, whereby solutions were inherited using both learning modes.

2.1.4 Types of Domain Knowledge Incorporated

In this section the incorporation of domain-specific knowledge in simple hybrids is examined. It is well established that the performance of hybrids correlates positively to the incorporation of domain knowledge, an outcome that is consistent with the "no free lunch theorem" [229]. A fundamental feature of hybrids is that the algorithms are typically designed to leverage or exploit any *a priori* knowledge on the inherent structure of the real-world problem. This problem-specific knowledge has been established to significantly enhance the performance of hybrids in solving the problem that they are designed for. The types of domain-specific knowledge that have been incorporated into hybrids to enhance search are in the form of specialized solution representations [162] and preprocessing schemes [87], as well as problem-specific search operators [3], etc.

Domain-specific solution representation permits flexible yet effective specialized operators to be exploited in the search [186]. The Genetic Vehicle Representation [162] was designed in the same spirit for solving capacitated vehicle routing problems. It is worth noting that domain-representation can also help narrow the search to a substantially smaller space compared to the original, where superior solutions are more likely to exist. Other successful specialized solution representations include the parallel job representation [130], the Artificial Chromosome with probability matrix [224], etc.

Domain-specific problem representation has also been used as a form of preprocessing in hybrids. In some algorithmic designs, domain-specific knowledge incorporated as part of the preprocessing steps, has been used to make the search more focused, thus enhancing the overall

search convergence rate. For instance, the pre-processing approach in [67], makes use of the *a priori* knowledge on redundant edges and nodes to reduce the problem size of the original generalized traveling salesman problem. Similarly, domain knowledge incorporated into fuzzy rule generation and prescreening prior to rule selection when dealing with high-dimensional pattern classification problems, has led to significant improvements in the efficiency of fuzzy rule selection using MOMA [77]. Other successful incorporations of domain knowledge for preprocessing include the block-cut tree techniques proposed in [90].

Taking advantage of the enriched representation, [6] describes a crafted crossover that exploits the specialized representation for effective exchange of partial routes between individuals, in vehicle routing problems. Specialized operators such as Position Based Mutator [227] have been proposed for solving job-shop scheduling problems more effectively. In many occasions, domain-specific information is also incorporated within the refinement process. Some of the notable schemes in single-objective, multi-objective optimization as well as uncertain, dynamic environments, include Concorde [7] which is designed for solving travelling salesman problem using the Brand & Cut method, the specially designed refinement operators [3] for accelerating the convergence rate to Pareto front, and the Random Multi-start Variable neighborhood Local Search [236], respectively.

Besides leveraging on domain knowledge in the design of specialized operators, the knowledge can be harnessed in other ways, such as within the fitness evaluation process. When handling problems with computationally expensive objective and constraints functions [151, 82, 68, 238], in particular, surrogate-based memetic algorithms or hybrids have been widely used [82, 185, 68, 26, 211] to enhance the computational tractability of the algorithm. In the refinement stage of such hybrids, approximation of the problem landscape is effected on the fly within a trust-region local refinement framework that manages the interplay between the original computationally intensive objective and constraint functions with computationally cheap surrogate models [211, 240, 115, 212]. With the zero-order and first-order consistency condi-

tions imposed at the initial guess, global convergence of the trust-region local search framework that embeds surrogate models can be guaranteed [4]. Hence surrogate-based memetic algorithm possesses the benefit of lower susceptibility in converging to false global optimum over other surrogate based evolutionary counterparts [173], since the use of surrogates are confined within the trust-region local search. Taking the spirit of optinformatics [110], a machine learning approach [69] to making geometrical and structural predictions on the feasibility of candidate solutions was introduced to effect the decision of performing further refinements on the individuals in the context of non-linear programming with active constraints.

2.2 Adaptive Hybrids

In recent decades, the general practice of hybridization in MAs has evolved into the emerging field of hybridization with adaptation. Adaptation of parameters and operators represents one of the most important and promising areas of research in memetic computation [149], as practitioners often encounter problems for which they have only limited insight into the structure of the problem, making it difficult to design specific MA without adaptation. These self-configuring algorithms are capable of acclimatizing to suit a given problem in hand, by methodically utilizing acquired information about the matching of problems to procedures, and reconfiguring itself to adapt to the problem as the search progresses. In contrast to simple hybrids in which domain knowledge is only captured and incorporated once by human expert during the design of MAs, adaptive algorithms handle the population diversity of the search and the various design issues by adaptable strategies and parameters for problem-solving [56, 149]. In this section the population diversity management and adaptation strategies for adaptive hybrids in memetic computation is presented.

2.2.1 Diversity Management

An important goal of adaptation in the design of search algorithms is the preservation of diversity in the solutions being explored. In single objective optimization, the problem of premature

convergence, whereby the population converges around a suboptimal solution, can be particularly problematic for hybrids. Premature convergence poses as a bigger challenge to hybrids in the contexts of multi-objective optimization and problems with uncertain and dynamic environments. In the former, a poorly spread Pareto front may result, while hybrids are more prone to failure in adapting to the changing environment once it has converged to some basin of attraction.

Population diversity refers to the extent of variation in the population based on the individuals' structure or performance [22]. Various measures of diversity have been used in MA, such as the extent of variations in fitness values [135], aging of solutions as defined in [37], etc. Based on these diversity measures, various techniques have been incorporated [134, 33, 214].

With fitness-based diversity, adaptive hybrids attempt to maintain the population diversity by managing good fitness spread among individual solutions in the population. In the coordination of multiple refinement procedures, [23] monitored how close the average fitness is compared to the population elite. Others, on the other hand, chose to effect the search adaptation according to the sparseness of individuals [214, 134] or the super-fit individual [25] in the population. To prevent loss in diversity, additional populations using completely different fitness criteria have also been considered [161]. When dealing with problems with uncertainties, population diversity is enhanced by locating and tracking multiple moving peaks via hierarchical clustering [232] and variable relocation evolutionary process [228] which makes already converged individuals evolve more rapidly to new environmental conditions. Other fitness-based diversity schemes proposed for single and multi-objective contexts include fitness sharing [33], clustering [94], adaptive grid [41], etc.

With distance-based diversity, adaptive hybrids attempt to maintain the population diversity by managing the distance spread among individual solutions in the population while the search progresses. Some relevant works in this discipline include adaptation based on population entropy [205], diversity-adaptive parallel memetic algorithm for solving large scale combinatorial

optimization problems [206], the many population management strategies proposed for handling multi-dimensional knapsack problems and weighted tardiness single-machine scheduling problems [195].

2.2.2 Design Adaptation Issues

Several core design adaptation issues of simple hybrids have been considered to date. In particular, it is now well-established [99] that potential algorithmic improvements can be achieved by adapting several key designs, including the frequency of refinement, selection of individual subset to undergo refinement, intensity of refinement, and choice of procedures to conduct refinement. In this section an overview of recent works directly on these issues is presented.

2.2.2.1 Frequency of Refinement and Individual Subset Selection

The frequency or probability of refinement defines the proportion of a population that should undergo the refinement process [198]. The aim is thus to balance the amount of computational budget allocated for population-based search versus refinement [70]. The adaptive selection of individual subset to undergo refinement helps enhance overall search productivity [138, 139, 99].

[78] investigated the impact of frequency of refinement in both single and multi-objective context for permutation flowshop scheduling, where only the elite individuals of the population undergo refinements. A fitness uniform selection scheme to adapt both the frequency of refinement and individual subset selection based on population diversity was also considered for cellular hybrids [138]. In particular, the trick lies in reducing the likelihood of applying the refinement procedure on individuals falling in previously refined basins of attraction.

2.2.2.2 Intensity of Refinement

Refinement intensity of an individual defines the amount of computational budget allocated to the refinement process. Study conducted to examine the effect of this design issue had

confirmed its significant impact on search performances [44].

The authors in [127] proposed an adaptive refinement procedure, whereby fitness statistics obtained from the population of individual are used to adaptively adjust the degree of refinement intensity. [128] adapts by building refinement chains beginning with a fixed intensity, and subsequently using these stored chains to define the degree of refinement intensity for new individuals. It is worth noting that an extended version of the local search chain memetic algorithm for high dimensional problems [129], i.e., 1000 dimensions, was demonstrated to top the chart in the recent competition at WCCI 2010. A crossover-based adaptive local search is also proposed in [142] which adapts the refinement intensity of the individual learning process based on some hill-climbing heuristic.

2.2.2.3 Refinement Procedures or Memes

In this subsection, the adaptive selection of refinement procedures or memes in memetic algorithm is reviewed. The choice of memes affecting the performance of hybrids significantly was demonstrated by several researchers on a variety of problems of diverse properties [149].

Several adaptation schemes have been put forward to adapt the choice of memes in hybrids. Hyper-heuristic [157] fuses a collection of memes in the form of low-level heuristics so that the actual meme applied on each individual may differ. It is a form of a heuristic to choose memes. Adaptive choice of memes at each decision point was also proposed in multi-meme algorithms [98] and meta-Lamarckian learning [149], while the co-evolution of memes and genes in adaptive MA was considered in [98]. [137] introduced a unified framework for evolutionary algorithm design that extends beyond the selection of memes to perform refinement. In particular, the probabilistic memetic framework described in [137] estimates the theoretical upper bound on computational budget for refinement as a means to control and adapt the multiple design issues of the hybrid simultaneously, at runtime, from how many and what individuals that would undergo lifetime learning to which appropriate meme(s) to employ in learning. To

alleviate the potentially high intensity and computational budget incurred in the refinement when solving problems with computationally expensive cost functions and uncertain environment, adaptive hill climbing [221] and trust-region based local searches [152, 82, 69] have also been introduced. Some of the refinement adaptation mechanisms proposed for handling MOO problems concentrated on the adaptive coordination of local refinement methods based on cross-dominance [24] and resource productivity criteria [21].

2.2.3 Theoretical Analysis of MA

Despite the broad research activities on the issues of MA design, there is the lack of sufficient rigorous theoretical study to date, since it is typically hard for theory to keep track with the state-of-the-art algorithms that has become increasingly elaborate and complex. Nonetheless, there are recent signs of increasing research in the design and development of formal analytical study on MA. *Polynomial local search complexity theory* [191], *fitness landscape analysis* [120, 182], *frequent pattern mining* [108], *connectivity structure analysis* [109], *local optima network* [219], *linkage graph* [181], *fixed-parameter algorithmics* [141] and *phenotypic plasticity* [158, 159, 20] are among some of those that have been introduced to facilitate our understandings of memetic algorithms, thus rendering the design of novel adaptive hybrid methodologies possible.

The polynomial local search (PLS) complexity theory was considered in [100] to analyze the worst case complexity of hybrids as a combination of population-based and refinement procedures, particularly in the discrete domain such as the Traveling Salesman Problem. The PLS completeness on a family of hybrids or memetic algorithms can be derived to gain insights on the expected worst case behavior of the solver on the problem of interest.

Fitness landscape analysis [120, 182], on the other hand, denotes one of the most popular tools for gaining insights into the problem fitness landscape with respect to the solver used. Local optima network [219] and connectivity structure analysis [109] represent some recent

theoretical analysis tools introduced to derive knowledge or more precisely the inherent properties of the search space, thus offering insights into the problem difficulty with respect to the hybridized.

In the same spirit, frequent schema mining [108], frequent subtree mining [86], and linkage graph discovery [181] were introduced for extracting latent solution patterns or recurring problem structure from online optimization data encountered along the search, thus enhancing the understanding on search dynamics and structural properties of the problem, thus improving search performances. Fixed-parameter algorithmics with kernelization techniques have also been proposed [141] for analyzing the given problem and divide the problem structure into smaller independent problem kernels that can be subsequently conquered and solved more efficiently.

On the theoretical forefront pertaining to mode of inheritance, [125] verified that the saddle-crossing ability of Baldwinian learning can help alleviate the issue of getting stuck in local optima, on simple multi-peaked and dynamic landscape. The higher short-term adaptability, phenotypic plasticity and other benefits of the Baldwin effect have also been studied in [158, 159, 20].

2.3 Concluding Remarks

Memetic computation is a broad generic framework that uses memes as units of information encoded in computational representations, and memetic algorithms as a whole is merely one aspect of the realization of memetic computation. This chapter has outlined several aspects of memetic computation consistent with research that has been carried out in the past. We expound on hybridization as being a foundational cornerstone for the application of memetic computation in problem-solving. This is especially relevant as the research community embarked on endeavors that resulted in many more techniques that surfaced onto the radar of optimization science in recent years. In this chapter, the capacity of memes in embracing

many of the techniques known and in the context of problem-solving, memetic computation will be a useful platform to capitalize on the richness of optimization tools known to practitioners. On the other hand, the diversity of memetic representational space explored to date for the constructions of notable memetic hybrids have remained insignificant. This necessitates the availability of a flexible memetic representations and frameworks that can conveniently accommodates learning and adaptation as well as evolution mechanisms that engage the principle of universal Darwinism. In the subsequent chapters, a flexible but under-exploited memetic representation, namely, *multi-memes* and *memeplexes*, and develop memetic frameworks that harness the property of meme co-adaptivity for the purpose of developing more effective problem-solving approaches in the context of memetic computation.

Chapter 3

Multi-Memes Individual Learning in Simple Hybrids

Many of the works in memetic computation take the form of simple hybrids, which combines population-based search with refinement procedures [99] and the incorporation of domain-specific knowledge [153]. In this chapter, a multi-memes individual learning for simple hybrids is investigated for CVRP, which serves as a refinement procedure, that involves multiple distinct memes combined in a synergistic manner for enhancing the problem-solving capabilities of a simple hybrid. The *Cooperating Memes Memetic Algorithm* (CMMA) which comprises of multiple memes within the individual learning is then proposed and studied in the context of CVRPs solving.

3.1 Introduction

Simple hybrids capitalize on the complementary advantage of population-based global search (explorative property) and local refinement (exploitative property) in that the former provides a reliable estimate of the global optimum while the latter concentrates the search effort around the best solutions found so far by searching their neighborhoods to produce better solutions more efficiently [149]. Consequently, the design of the refinement procedure greatly affects the convergence reliability and efficiency of the simple hybrid. Numerous local search algorithms

or memes have been proposed as refinement procedures in simple hybrids for the past decades. On the other hand, as emphasized earlier in Chapter 2 and according to the well-known *No Free Lunch* theorem for search [229], no single meme or optimization method always works best on all problems [229]. Besides varying greatly from one problem instance or domain, the refinement performance of any meme could vary throughout different stages of a search. Thus, there is an inherent risk associated with choosing a single meme or local search in a simple hybrid for solving a given problem, since in reality no a priori knowledge is often available to assess which meme is most suitable for the problem instance or across the entire search. In some experiments, it is even known that different local search or individual learning operators are better during different phases of the population-based search [214, 134, 33, 214].

In practice, a simple hybrid that works well across a large variety of problem instances of differing characteristics would be desirable for solving real-world applications where prior knowledge is often unavailable, as opposed to a simple hybrid with a single meme that performs exceptionally well on particular problem instances but poorly on others. This chapter proposes a *Cooperating Memes Memetic Algorithm* (CMMA), that combines memes of cooperating nature in the individual learning of memetic search. This reduces the risks of failure by the simple hybrid on problems of differing properties while improving search efficacies in solution quality, efficiency and robustness. The basic idea of CMMA can be described as follows: instead of putting all hopes on a single meme for local refinement, the same local search time budget is on the other hand invested across multiple memes. Moreover, the synergy between multiple simple memes can lead to notable complex memes with emergence search behaviours that excel in search [72]. In essence, CMMA comprises a multi-memes individual learning that capitalizes on the unique search biases of each meme as well as their synergistic behaviours for more effective problem solving. To demonstrate the efficacy of the proposed CMMA, several sets of Capacitated Vehicle Routing Problems (CVRPs) of various characteristics are used as benchmarks on which the performances of CMMA can be assessed, validated and analyzed.

In what follows, Section 3.2 begins with the conceptualization of the multi-memes individual learning in *Cooperating Memes Memetic Algorithm*. Section 3.3 presents a literature review of the well established vehicle routing problems, with specific focus placed on the CVRPs and its formulations. Section 3.4 describes the multi-memes individual learning considered, including the terminal set of diverse memes used for solving CVRP. A depiction on the workflow of CMMA is also included in the section. The empirical study on effectiveness and efficiency of the proposed CMMA is then presented in Section 3.5. The main conclusions are subsequently drawn in Section 3.6.

3.2 Multi-Memes Individual Learning

The outline of canonical simple hybrid is as illustrated in Alg. 1. The canonical memetic algorithm or simple population-based with individual learning hybrid begins its search with the initialization of a population $\text{Pop}(g = 0)$ of candidate solutions. At each generation g , a population of offspring $\text{Pop}(g) = \{s_1, s_2, \dots, s_M\}$ is generated using population-based search methodologies such as evolutionary computation through its evolutionary operators like crossover and mutation. $\text{Pop}(g)$ then undergoes fitness evaluation and individual learning (in which solutions $s_i \in \text{Pop}(g)$ are refined with a local improvement procedure or otherwise often labelled as a meme m). This workflow of memetic search repeats until the stopping criteria defined are reached.

Algorithm 1 Outline of Canonical Simple Hybrid

Input: Problem Instance**Output:** Global best solution s_g *Create* an initial population of solutions $\text{Pop}(g = 0) = \{s_1, s_2, \dots, s_i, \dots, s_M\}$ **while** stopping conditions are not satisfied **do** $g \leftarrow g + 1$ **Perform** evolutionary operators to generate $\text{Pop}(g)$ from $\text{Pop}(g - 1)$ based on $f(s_i)$ **Evaluate** $f(s_i), \forall s_i \in \text{Pop}(g)$ **for** $s_i, \forall s_i \in \text{Pop}(g)$ **do** $s_i \leftarrow \text{IndividualLearning}(s_i, m)$ **end for****Update** global best solution s_g **end while**

The individual learning procedure of simple hybrids on discrete problems, a meme m often denotes a local heuristic search method¹ that can be used to refine the solution s produced by the population-based search process of the hybrid. Each meme thus encodes the unit of instruction for performing local refinement during the individual lifetime learning process. Without loss of generality, a meme m in individual learning can be viewed as imposing a particular neighborhood that can be searched upon. The factors that influence the efficacy of local search can be delineated by the move strategy, neighboring solution acceptance strategy, search depth, and expressed in the tuple form:

$$m = (\langle T_i \rangle_m, \langle \text{Acceptance_Strategy} \rangle_m, \langle \text{Search_Budget} \rangle_m) \quad (\text{Eq. 3.1})$$

where $\langle T_i \rangle_m$ denotes a move operator that defines the search neighborhood structure of m , $\langle \text{Acceptance_Strategy} \rangle_m$ defines the criteria for accepting a solution, such as *first improve-*

¹A heuristic is an experience-based technique designed for problem solving that usually produces a good solution.

ment or best improvement, while $\langle Search_Budget \rangle_m$ defines the local search time budget incurred by meme m , which is usually specified based on maximum number of local refinement iteration. Alg. 2 depicts a canonical individual learning procedure where a meme m is mobilized for refinement of solution s_i .

Algorithm 2 Outline of IndividualLearning(s_i, m)

Input: s_i, m

Output: s (solution refined)

$s \leftarrow s_i$

$t \leftarrow 0$

while $t \leq \langle Search_Budget \rangle_{m_l}$ **do**

/ Refine(s, m_l) */*

$s' \leftarrow \text{SearchNeighborhood}(s, \langle T_i \rangle_m, t)$

if $\text{Accept}(s', s, \langle Acceptance_Strategy \rangle_m)$ **then**

$s \leftarrow s'$

end if

$t \leftarrow t + 1$

end while

Return s

The individual learning procedure based on a single meme m , as illustrated in Alg. 2, is limited in several aspects. First of all, there is the inherent risk associated with choosing a single meme for individual learning in underperforming within the simple hybrid search on a given problem of interest, due to the intrinsic bias of each whose meme will carry search mechanism which varies greatly from one problem class to another, as well as throughout different stage g of the entire search. Furthermore, in spite of many existing memes that have been used for solving real-world problems, it is generally unknown in advance which meme is most appropriate for a problem or class of problems and whether it serves well throughout the different stages of the simple hybrid search even on a particular problem class. In many cases,

individual learning based on single meme often gets trapped in some local optima after some refinement steps, thus making it ineffective.

To address the above mentioned issues associated with present individual learning procedure that employs single meme, this work considers a multi-memes individual learning procedure, capable of tapping on the diverse domain-driven biases inherent to unique memes, in a simple hybrid. In the simplest form, the time budget previously allocated to single meme is now invested across multiple memes $\mathcal{M} = \{m_1, m_2, \dots, m_l, \dots, m_L\}$, as illustrated in Alg. 3.

Algorithm 3 Outline of IndividualLearning($s_i, \mathcal{M}$)

Input: $s_i, \mathcal{M}$

Output: s (solution refined)

$s \leftarrow s_i$

$t \leftarrow 0$

for $m_l \in \mathcal{M}$ **do**

while $t \leq \langle Search_Budget \rangle_{m_l}$ **do**

/ Refine(s, m_l) */*

$s' \leftarrow \text{SearchNeighborhood}(s, \langle T_i \rangle_{m_l}, t)$

if $\text{Accept}(s', s, \langle Acceptance_Strategy \rangle_{m_l})$ **then**

$s \leftarrow s'$

end if

$t \leftarrow t + 1$

end while

end for

Return s

Fig. 3.1 gives a detailed depiction that compares and contrasts the schematic workflow between a simple hybrid with single meme individual learning to a multi-memes individual learning. As illustrated in Fig. 3.1, both hybrids incur similar computational resources by

maintaining $\langle Search_Budeget \rangle_m \simeq \langle Search_Budget \rangle_M$. On the other hand, while *single-meme* individual learning is more likely to get trapped in local optima, the proposed multi-memes individual learning is capable of jumping out of one basin of attraction to another due to the presence of diverse memes that works together in a *cooperative* manner (and under the same time budget) under a shared goal of finding the global optimum.

Fig (a) Single-Meme Individual Learning

Fig (b) Multi-Memes Individual Learning

Figure 3.1: Workflows of Single-meme Individual-Learning versus Multi-memes Individual-Learning

3.3 Capacitated Vehicle Routing Problem

In this section, the *Vehicle Routing Problem* (VRP) is described. The VRP [36] is a widely studied [103] NP-hard [71] combinatorial optimization problem² that was introduced in 1959 by Dantzig and Ramser [38]. The problem involves finding a set of optimal routes starting and ending at a single depot that together visit a set of customers [103]. In VRP, a set of N customers $\mathcal{E}_{\mathcal{N}}$ is to be served by a fleet of K vehicles $\mathcal{V}_{\mathcal{K}}$

$$\mathcal{E}_{\mathcal{N}} = \{e_1, e_2, \dots, e_l, \dots, e_N\} \quad (\text{Eq. 3.2})$$

$$\mathcal{V}_{\mathcal{K}} = \{v_1, v_2, \dots, v_k, \dots, v_K\} \quad (\text{Eq. 3.3})$$

Each customer $e_l \in \mathcal{E}_{\mathcal{N}}$ must be serviced by one vehicle $v_k \in \mathcal{V}_{\mathcal{K}}$. Each vehicle v_k starts and ends at the central depot e_0 , and travels a route τ_k as given by:

$$\tau_k = \{a_{k1}, a_{k2}, \dots, a_{kj}, \dots, a_{kn}\} \quad (\text{Eq. 3.4})$$

where a_{kj} represents the j^{th} customer visited by v_k and $n = |\tau_k|$ denotes the number of customers visited by v_k .

The objective of VRP is to find the optimal routing solution $s = \{\tau_1, \tau_2, \dots, \tau_i, \dots, \tau_K\}$ that minimizes the overall distance traveled by all vehicles. Variants of VRPs include different constraints such as capacity constraint, maximum route length constraint, time window constraint, multi-depot constraint, etc. The *Capacitated Vehicle Routing Problem* (CVRP) is the capacitated variant of VRP that involves additional constraints such as capacity and max route length constraints. In a CVRP, each customer e_l has $\text{demand}(e_l)$ that must be serviced by the visiting vehicle v_k , and each vehicle v_k has a finite capacity denoted as C_v . The objective of the CVRP is to find an optimal routing solution s_{opt} , that minimizes the overall distance traveled

²Combinatorial optimization problem refers to an optimization problem in which feasible solutions are discrete or can be reduced to discrete [57].

by all vehicles, under the strict constraints that each customer is serviced exactly once by a single vehicle, and the total demands of all customers serviced by any vehicle v_k does not exceed vehicle capacity C_v . In addition, each vehicle may travel a maximum allowable distance of L_{max} .

Vehicle routing problem and its variants such as CVRP represent the cornerstone of optimization for distribution networks as routing decisions must be made in essentially every distribution network. Improved vehicle routing methods represent a significant savings potential to businesses throughout the world. This importance is highlighted by the scale of modern distribution networks. For instance, [1] reported that the cost of logistics in the US was estimated at \$1.1 trillion in 2009, while the cost of logistics in China was estimated at 3.8 trillion yuan in 2006 and growing at an extraordinary rate. Moreover, the growth of distribution network has a profound impact on the environment, at the time when the transport sector is reported to be the fastest growing source of greenhouse gases, and from which over 85% of total greenhouse gas emission is contributed by road vehicles CO_2 emissions [146]. For that reason, even small improvements introduced by vehicle routing methods into the distribution network, can yield significant savings and have a potentially huge impact on the world environment such as green gas emission.

3.3.1 Problem Formulation

A solution s in VRP defines a set of K routes:

$$s = \{\tau_1, \tau_2, \dots, \tau_k, \dots, \tau_K\} \quad (\text{Eq. 3.5})$$

where τ_k denotes the route visited by vehicle v_k , and K is the number of vehicles in s_i .

Route τ_k defines an ordered set of customers:

$$\tau_k = \{a_{k1}, a_{k2}, \dots, a_{kj}, \dots, a_{kn}\} \quad (\text{Eq. 3.6})$$

where a_{kj} denotes the j^{th} customer on route τ_k visited by v_k and $n = n(k) = |\tau_k|$ is the number of customers visited by vehicle v_k .

The inclusion of depot e_0 is enforced in each route τ_k (i.e., $e_0 = a_{kj}, \exists a_{kj} \in \tau_k$). Hence, every route τ_i forms a closed TSP circuit. Fig. 3.2a illustrates an example of the VRP and the respective multi-route solution representation is given in Fig. 3.2b. Route $\tau_1 = \{e_7, e_{11}, e_0, e_2, e_5\}$ in Fig. 3.2b denotes the *closed circuit* $e_0 \rightarrow e_2 \rightarrow e_5 \rightarrow e_7 \rightarrow e_{11} \rightarrow e_0$ visitation route of vehicle v_1 .

Figure 3.2: Multi-Route VRP Solution Representation

The distance traveled by vehicle v_k is denoted here as $L(\tau_k)$ and is given by:

$$L(\tau_k) = \sum_{k=1}^{n-1} d(a_{kj}, a_{k(j+1)}) + d(a_{k1}, a_{kn}) \quad (\text{Eq. 3.7})$$

where $d(a_{kj}, a_{k(j+1)})$ represents the distance between two consecutive customers a_{kj} and $a_{k(j+1)}$ on route τ_k .

On the other hand, the total demand serviced by vehicle v_k is denoted as $D(\tau_k)$ and is given by:

$$D(\tau_k) = \sum_{k=1}^n demand(a_{kj}) \quad (\text{Eq. 3.8})$$

where $demand(a_{kj})$ corresponds to the demand of customer a_{kj} .

The objective of the CVRP is to minimize the overall distance $C(s)$ traveled by all K vehicles and is defined as:

$$C(s) = \sum_{k=1}^K L(\tau_k) \quad (\text{Eq. 3.9})$$

$$= \sum_{k=1}^K \left[\sum_{j=1}^{n-1} d(a_{kj}, a_{k(j+1)}) + d(a_{k1}, a_{kn}) \right] \quad (\text{Eq. 3.10})$$

subjected to constraints:

$$\begin{aligned} L(\tau_i) &\leq L_{max} \quad \forall \tau_i \in s \\ D(\tau_i) &\leq C_v \quad \forall \tau_i \in s \end{aligned} \quad (\text{Eq. 3.11})$$

where L_{max} is the maximum allowable distance traveled by each vehicle, and C_v is the capacity of each vehicle.

Some commonly used CVRP benchmark problems already extensively discussed in the literature are used here in this research work. The inclusion of a multitude of benchmark classes with wide-ranging characteristics make it possible to analyze the proposed CMMA in comparison with existing state-of-the-art approaches. They represent CVRP classes that differ in size and other characteristics [42], including customer distribution, symmetric structure, route-length restriction, and vehicle capacity.

- Augerat: This set of benchmarks was proposed by Augerat et al. [8]. We considered 8 well-known Augerat benchmarks including 5 random instances from Augerat set A with node size ranging from 32 to 80 and 3 clustered instances from Augerat set B with node size ranging from 57 to 78.
- CE: This set of benchmarks was proposed by Christofides and Eilon [31]. The 4 benchmark instances are modified versions of a 76 customer instance that differ in vehicle capacity.
- Christofides: This set of benchmarks was proposed by Christofides et al. [32]. The node size of the problem instances range from 50 to 199 customers. In addition, 7 instances enforce maximum route length constraint.

- Golden: This set of benchmarks was proposed by Golden et al. [63]. The set contains 20 large-scale VRP instances with customer size ranging from 200 to 483. The large instances are designed to contain highly symmetrical structures. In instances 1-8, maximum route length constraint is applied, and the nodes are located in concentric circles around the depot.

3.3.2 Literature Review

Being one of the most important practical problems of Operational Research, VRP and its variants are also considered as one of the most difficult problems due to its complex combinatorial nature; it is the fusion of two NP-hard problems, namely the Traveling Salesman Problem (TSP) and the Bin Packing Problem (BPP). For VRP instances with few nodes, the branch and bound method is deemed to be effective and known to provide the best solution to date [10]. However, the exponential growth of the difficulty of the problem as its size and complexity increases makes exact methods such as branch and bound not viable for large-scale VRP, and classical VRP of medium scale dimension remains computationally intractable using the exact enumeration methods[215]. As a result, most researchers have turned to metaheuristics and simple hybrids in memetic computation for solving real world VRPs. Genetic algorithms [81, 222, 168], Simulated Annealing (SA) [32, 156, 5], Tabu Search (TS) [32, 58], Ant Colony Optimization [55], and constraint programming [189], represent some of the popular metaheuristics algorithms that have been developed for handling real-life VRPs, with reasonably high degree of success. Gendreau et al. [59] reported that Tabu Search (TS) with specialized mechanisms and data structures were among some of the most successful metaheuristic algorithms currently available for dealing with large-scale VRPs. Evolutionary algorithms (EAs) such as GAs have also shown very promising results in real-world VRPs such as Capacitated VRPs [230], VRPs with Time-varying Travel Times [81], VRP disruption management with Fuzzy Time Window [223] or the Request Changes of Customers [222], etc.

It has been noted in [36] which assessed ten VRP metaheuristics published after 2000 on two sets of classical VRP benchmarks that tabu search methods have now been superseded by simple hybrids in memetic computation. In fact, for the CVRP benchmark set with large instances (Golden et al. [63], 200 to 483 customers), the methods shown to be able to converge within 1% to best-known solutions are all simple hybrids in memetic computation, including Active Guided Evolution Strategy (AGES) of Mester and Bräysy [121], the D-Ants algorithm of Reimann et al. [175], the memetic algorithm (MA) of Prins [170] and the Bone Route adaptive memory based procedure of Tarantilis and Kiranoudis [209], as well as two more heuristics recently added to this 1% club by Tarantilis [208] and Pisinger and Röpke [164]. Other notable simple hybrids include the Genetic Vehicle Representation (GVR) proposed by Pereira et al. [162], the fitness landscape and distance metrics, the memetic hybrids with distance-preserving recombination operators by Marek Kubiak [101, 102], the co-evolutionary memetic algorithm for minimizing the total traveled distance by Berger and Barkaoui [11], and etc.

3.4 Cooperating Memes Memetic Algorithm for solving CVRP

This section describes the *Cooperating Memes Memetic Algorithm* (CMMA) [27], which is a simple hybrid with a multi-memes individual learning designed for solving CVRP. The basic steps of the proposed CMMA are outlined in Alg. 4. In the first step, the initial population $\text{Pop}(g = 0)$ of candidate solutions for the CVRP is created either randomly or using the sweeping constructive heuristic [60]. The initial population then iteratively undergoes evolution with the reproduction operations, namely crossover and mutation, followed by a multi-memes individual learning to arrive at a population of local optima. A *split-repair* operator is also designed to recover any infeasible solutions produced by the genetic operators. This workflow of the memetic search proceeds until the computational budget is exhausted or a user-specified termination criterion is met. The multi-memes individual learning operator $\mathcal{M}$ consists of 6

different memes that refines solutions in $\text{Pop}(g)$ in three stages respectively. In the rest of this section, the *VRP-specific* evolutionary operators and *split-repair* are described in great details.

Algorithm 4 Outline of CMMA Workflow

Input: Problem Instance

Output: Global best solution s_g

Create an initial population of solutions $\text{Pop}(g = 0) = \{s_1, s_2, \dots, s_i, \dots, s_M\}$

while stopping conditions are not satisfied **do**

$g \leftarrow g + 1$

Perform evolutionary operators to generate $\text{Pop}(g)$ from $\text{Pop}(g - 1)$ based on $f(s_i)$

Evaluate $f(s_i), \forall s_i \in \text{Pop}(g)$

/*Multi-memes Individual-Learning*/

for $s_i \in \text{Pop}(g)$ **do**

repeat

Perform Individual-Learning on s_i using the memes m_1, m_2, m_3 in stage 1

Perform Individual-Learning on s_i using the memes m_4 in stage 2

Perform Individual-Learning on s_i using the memes m_5, m_6 in stage 3

until local search time_budget(multi-memes-individual-learning) is exceeded

end for

$\text{Pop}(g) \leftarrow \text{Merge}(\text{Pop}(g), \text{Pop}(g - 1))$

Update global best solution s_g

end while

3.4.1 Population-based Evolutionary Search

In CMMA, population-based evolutionary operators are applied to $\text{Pop}(g)$ to produce the next generation of solutions $\text{Pop}(g + 1)$. These reproduction operators mainly consists of crossover, mutation, and a split-repair operator to revive infeasible solution from crossover and mutation operators. As multi-memes individual learning is a computationally intensive process. the solution population $\text{Pop}(g)$ is kept as small.

3.4.1.1 Crossover

For crossover between solutions s_i and s_j , the routes in child solution s_c are generated from partial routes of parent solutions s_i, s_j . In particular, taking the depot node e_0 as the axis center, a bisecting axis ψ of random angle θ is generated to slice the VRP graph into two separate subgraphs containing non-empty customers, H_+ and H_- , and the child solution s_c is a fusion of partial routes in s_i and s_j (e.g. In Fig. 3.3, s_c is created from $\{\tau_1^i, \tau_2^i, \tau_3^i\} \in s_i$ and $\{\tau_1^j, \tau_3^j, \tau_4^j\} \in s_j$). The angle θ is selected in such a way as to ensure that both subgraphs are non-empty [27].

Figure 3.3: Genetic Crossover between s_i and s_j (phenotype)

3.4.1.2 Mutation

The mutation operator is performed by one of the four mutation heuristics, namely *swap*, *inversion*, *insertion* and *displacement*, through random selection, which is purely stochastic in nature and each heuristic shares equal probability in the mutation of a solution s_i .

The mutation level p_m and the size M of the genetic population $\text{Pop}(g)$ affects the overall convergence reliability and velocity of the proposed CMMA [177, 167]. In general, if the mutation is too weak, solution s will get stuck in the attraction basin of local optimum s' . On the other hand, if it is excessive, s and s' are unrelated and evolutionary search will behave more like a greedy randomized adaptive search procedure (GRASP) [145]. In VRP which is characterized by combinatorially explosive solutions, small values of M (e.g., $M \geq 15$) are not sufficient to handle the amount of information included into the individual solutions. On the other hand, although large population size (e.g. $M \leq 50$) fully exploit the large diversity of permutation, this is made at the expense of convergence velocity.

For CMMA solving combinatorially explosive problems such as CVRP, the population size is kept small at $M = 30$ (e.g., as in [177]) to enhance the convergence speed of CMMA which already has a computationally intensive multi-memes individual learning. On the other hand, in contrary to classical genetic algorithms where mutation is applied at low rates, the CMMA mutation operator is typically applied at high rates $p_m = 0.5$ [167] to increase population diversity and hence the population dynamics for a small population.

3.4.1.3 Split-Repair

Due to the VRP constraints in Eq. 3.11, the mutation and crossover operators may generate infeasible solutions. In CMMA, a *split-repair* procedure is designed to revive any infeasible solution. For an infeasible solution s_{inf} , some vehicle(s) v_k either violate the capacity constraint or distance constraint or both. Let

$$\Gamma = \{\tau_i | \tau_i \in s_{inf} \text{ and } \tau_i \text{ represents an infeasible route}\}$$

be a set of infeasible routes in solution s_{inf} . Thus $\Gamma' = \{\tau_i | \tau_i \in s_{inf} \text{ and } \tau_i \notin \Gamma\}$ is the set of feasible routes in s_{inf} . In this procedure, an infeasible route τ_i is removed from Γ and split into two routes τ_a and τ_b by removing the edge between consecutive customers a_{ik} and $a_{i(k+1)}$ such that the increase $\delta(k)$ in overall distance traveled due to the *split* is minimum.

Algorithm 5 Outline of *Split-Repair(s)* procedure

Input: s (infeasible solution)**Output:** s (feasible solution)Split s into Γ, Γ' **while** $\Gamma \neq \emptyset$ **do** Given $\tau_i \in \Gamma$ **for** $j = 1$ to m **do** $\delta(j) =$ distance increase by splitting τ_i between a_{ij} and $a_{i(j+1)}$ **end for** $k \leftarrow j$ with $\min\{\delta(j)\}$ **Split** τ_i between a_{ik} and $a_{i(k+1)}$ to form τ_a and τ_b **for** $\tau^* \in [\tau_a, \tau_b]$ **do** **if then** τ^* is infeasible **then** $\Gamma \leftarrow \Gamma + \tau^*$ **else** $\Gamma' \leftarrow \Gamma' + \tau^*$ **end if** **end for** $\Gamma \leftarrow \Gamma - \tau_i$ **end while**

3.4.2 Multi-Memes Individual Learning for CVRP

Here the multi-memes individual learning, which harnesses the unique domain-driven search biases of multiple memes during individual learning, is presented with respect to solving CVRPs. The multi-meme individual learning procedure is constructed from a terminal set of memes for the CVRP, $\mathcal{M} = \{m_1, m_2, \dots, m_6\}$. The proposed multi-memes individual learning involves $m_1 \dots m_6$ hybridized and organized in a *cooperative* manner (e.g. Clustering $\rightarrow$ Merging $\rightarrow$ TSP Route Optimization), which allows these memes to work together to accomplish the shared optimization goal.

3.4.2.1 Individual-Learning Meme Terminal Set for CVRP

The meme terminal set for CVRP is constructed from 6 commonly used move operators for VRP solution refinement, $\{T_1, T_2, \dots, T_6\}$. They are depicted in Fig. 3.4 and Fig. 3.5. Move

operators T_1, T_2, T_3 are clustering procedures to reassign customers between routes so as to reduce the overall traveled distance. T_4 denotes a move operator that reduces the total number of routes in the current solution by route merging. Move operators T_5, T_6 , on the other hand, reduce the route length of a single route represented in the current solution by reassigning the visiting order of customers in the route. In what follows, each of the move operators are briefly described:

T_1 : customer τ_{kj} of route $\tau_k \in s$ is removed and reinserted into a separate route $\tau_{k'} \in s, \tau_{k'} \neq \tau_k$, if fitness improvement $\Delta C(s) > 0$.

T_2 : customer τ_{kj} of route $\tau_k \in s$ is swapped with customer $\tau_{k'l}$ of route $\tau_{k'} \in s, \tau_{k'} \neq \tau_k$, if fitness improvement $\Delta C(s) > 0$.

T_3 : a sub-route $\hat{\tau}_k = \{\tau_{k(j+1)}, \tau_{k(j+2)}, \dots, \tau_{k(j+\mu)}\}$ of route $\tau_k \in s$ is swapped with sub-route $\hat{\tau}_{k'} = \{\tau_{k'(l+1)}, \tau_{k'(l+2)}, \dots, \tau_{k'(l+\nu)}\}$ of route $\tau_{k'} \in s, \hat{\tau}_k \neq \hat{\tau}_{k'}$, if fitness improvement $\Delta C(s) > 0$.

T_4 : two shorter routes $\tau_k, \tau_{k'}$ in the current solution are merged to reduce the number of routes and thus the overall distance traveled, if fitness improvement $\Delta C(s) > 0$.

T_5 : a customer $\tau_{kj} \in \tau_k$ is reinserted after a customer $\tau_{kl} \in \tau_k, \tau_{kj} \neq \tau_{kl}$, if fitness improvement $\Delta C(s) > 0$.

T_6 : a customer $\tau_{kj} \in \tau_k$ is joined with a customer $\tau_{kl} \in \tau_k, \tau_{kl} \neq \tau_{kj}$ by reversing the visiting order of customers between τ_{kj} and τ_{kl} , if fitness improvement $\Delta C(s) > 0$.

 Figure 3.4: Move Operators: T_1, T_2, T_3

 Figure 3.5: Move Operators: T_4, T_5, T_6

3.4.2.2 Workflow of Multi-Memes Individual Learning for CVRP

The complete multi-memes individual learning operator for solving CVRP is outlined in Alg. 6. As illustrated in Alg. 6, the proposed individual learning procedure is divided into three separate stages, where each stage possesses unique learning roles and search properties, is next described.

Algorithm 6 Outline of Multi-Memes Individual-Learning for solving CVRP

Input: $\text{Pop}(g)$ **Output:** refined solutions $\{s | s \in \text{Pop}(g)\}$ **for** $s \in \text{Pop}(g)$ **do** */* Stage 1: Clustering */* **repeat** $s' \leftarrow \text{reallocate}$ customers between $\tau_i \in s$ and $\tau_k \in s$ using m_1, m_2, m_3 that maximizes $\Delta C = C(s) - C(s')$ **if** $\Delta C > 0$ **then** $s \leftarrow s'$ **end if** **until** $C(s) \leq C(s'), \forall s' \in \mathcal{N}(s)$ */* Stage 2: Routes reduction */* **repeat** $s' \leftarrow \text{merge}$ $\tau_i \in s$ and $\tau_k \in s$ using m_4 that maximizes $\Delta C = C(s) - C(s')$ **if** $\Delta C > 0$ **then** $s \leftarrow s'$ **end if** **until** $C(s) \leq C(s'), \forall s' \in \mathcal{N}(s)$ */* Stage 3: TSP Optimization */* **for** $\tau_i \in s$ **do** **repeat** $s' \leftarrow \text{re-order}$ customers in τ_i using m_5 and m_6 that maximizes $\Delta C = C(s) - C(s')$ **if** $\Delta C > 0$ **then** $s \leftarrow s'$ **end if** **until** $C(s) \leq C(s'), \forall s' \in \mathcal{N}(s)$ **end for** $s \leftarrow s'$ if s' is fitter than s in the spirit of *Lamarckian Learning***end for**

- Stage 1 individual learning involves a clustering refinement process to improve customer allocation between routes so as to reduce the overall traveled distance. The composite edge-exchange neighborhood search strategy is composed of three memes: meme m_1

with T_1 move operator for *customer relocation*, meme m_2 with T_2 move operator for *customer exchange*, and meme m_3 with T_3 for *route exchange*.

- Stage 2 individual learning involves the routes merging meme m_4 , which serves to reduce the number of vehicles as well as the overall distance traveled using memes via the T_4 move operator.
- Stage 3 individual learning involves the TSP optimization to improve the solution quality of each route. The composite edge-exchange search neighborhood is composed of two memes: meme m_5 with move operators T_5 for *reinsertion* and m_6 with T_6 for *2-exchange*.

The organization of stage 1, 2 and 3 as well as the memes 1-6 is based on the domain-based observation for CVRP in which local refinement performs more efficiently when it is designed to refine solution in a gradually more granular level, i.e. from more high-level refinement (inter-route refinement, as performed by memes m_1, m_2, m_3) to medium-level refinement (route-merging refinement, as performed by meme m_4) and to low-level refinement (refinement within each individual route, as performed by memes m_5, m_6).

3.4.2.3 Computational Complexity of Multi-Memes Individual-Learning for CVRP

As memes for CVRP are typically computationally intensive heuristics, in order to improve the overall complexity of proposed algorithm, several speed-up schemes are introduced in CMMA to accelerate examinations of neighborhood without loss in solution quality. Firstly, when handling edge-exchange move operators such as T_1, T_2, T_3, T_5, T_6 , instead of evaluating the objective function for each neighboring solution (which has a computational complexity of $O(N)$, where N is the number of customers in a CVRP instance), CMMA only keeps track of the differences with respect to the edges that have been deleted or added. Furthermore, instead of evaluating the entire neighborhood at each individual learning step, only exchanges involving modified routes are re-evaluated once T_i executes a move, since the effect of such

edge-exchanges is limited to the routes been modified. Lastly, CMMA only remembers the top few non-tabu move. All these schemes effectively reduces the computational complexity of the neighborhood examination to $O(1)$.

3.5 Empirical Study

In this section, the empirical study on the proposed CMMA for solving the CVRPs is presented. In the present study, A diverse set of CVRP benchmarks as described in Subsection 3.3.1 are used here. These CVRP benchmarks differ in size and other properties [42], including customer distribution, symmetric structure, route-length restriction, and vehicle capacity. The inclusion of a multitude of benchmarks with wide-ranging characteristics thus makes it possible to assess, validate and analyze the proposed CMMA in comparison to existing state-of-the-art approaches. To study and analyze the efficiency and effectiveness of the proposed CMMA, a diverse set of state-of-the-art metaheuristics and memetic hybrids are used as baseline for comparison on the representative benchmark problems. In particular, the set consists of variants of i) VRP domain-specific Memetic Algorithms ii) Simulated Annealing iii) Tabu Search iv) Genetic Algorithms and v) VRP domain-specific Local Search, as candidate stochastic search baselines, with a total of 15 stochastic search algorithm instances considered and listed as follows:

These metaheuristics and simple hybrids chosen for comparison form the set of recent approaches that have either reported competitive results or new best known solutions on the representative benchmark problems used in the present study, which includes both the recently introduced memetic algorithms for solving the large-scale Golden benchmark set. (i.e., *BR* [209] and *SEEA* [170])

<i>SEEA</i>	-	Memetic Algorithm proposed by Prins in [170] that uses specialized splitting procedure in place of trip delimiters.
<i>BB</i>	-	<i>Co-evolved Hybrid Genetic Algorithm</i> proposed in [11]
<i>OS</i>	-	Metastrategy Simulated Annealing algorithm proposed in [156]
<i>GHL</i>	-	<i>TabuRoute</i> proposed in [58]
<i>CGL</i>	-	<i>Tabu Search</i> proposed in [35]
<i>GTS</i>	-	<i>Granular Tabu Search</i> proposed in [216]
<i>RR</i>	-	<i>Parallel Tabu Search Algorithm</i> using <i>Ejection Chain</i> proposed in [174]
<i>BR</i>	-	<i>Boneroute</i> Memetic Algorithm proposed in [209]
<i>XS</i>	-	<i>Xu and Kelly Tabu Search</i> proposed in [231]
<i>GTS</i>	-	<i>Granular Tabu Search</i> proposed in [216]
<i>GVR_x</i>	-	Genetic Algorithm with <i>Genetic Vehicle Representation</i> proposed in [162]
$(x = 1, 2, 3, 4)$		
<i>RTR</i>	-	<i>Record-to-Record Algorithm</i> proposed in [63]
<i>RT</i>	-	<i>Probabilistic Diversification and Intensification Local Search</i> proposed in [180]
<i>TA</i>	-	<i>Parallel Iterative Search Method</i> proposed in [200]
<i>WH</i>	-	<i>Repeated Matching Heuristic</i> proposed in [225]

The search quality of the CMMA was revealed through close examinations on the performances of the metaheuristics under comparison as tabulated in Tables 3.1, 3.2, 3.3, 3.4, and 3.5. In these tables, column 1 denotes VRP instance with n denoting the problem size (i.e. the number of customers serviced). The remaining columns tabulate the performances of metaheuristic algorithm in terms of the following metrics:

These metrics encompass the common criteria to facilitate possible and fair comparison between CMMA and other state-of-the-art metaheuristic and memetic algorithm on the representative benchmark problem instances. All the results presented for CMMA are collected over 30 independent runs for each VRP instance on a common computing platform of Pentium-4 2.66 GHz with 1 GB RAM. Each run continues until a maximum of 300,000 fitness evaluations is reached or the solution converged to the best result known in the literature. The population-based search parameters of the simple hybrid for solving the benchmark problems were configured as follows: crossover and mutation probability of $p_c = 0.5$, $p_m = 0.5$, roulette-wheel

Table 3.1: Results on Augerat benchmarks (The best performing algorithm of the respective CVRP instance is highlighted in Bold)

$i(n)$	k	$\bar{C}$	$\overline{gap}$	C	gap	C_{best}
A32k5(32)	<i>CMM A</i>	784.00	0.00%	784.00	0.00%	784
	<i>GV R</i> ₁	793.40	1.20%	784.00	0.00%	
	<i>GV R</i> ₂	790.90	0.88%	784.00	0.00%	
	<i>GV R</i> ₃	796.50	1.59%	784.00	0.00%	
	<i>GV R</i> ₄	786.30	0.29%	784.00	0.00%	
A54k7(54)	<i>CMM A</i>	1168.37	0.12%	1167.00	0.00%	1167
	<i>GV R</i> ₁	1178.60	0.99%	1167.00	0.00%	
	<i>GV R</i> ₂	1186.20	1.65%	1167.00	0.00%	
	<i>GV R</i> ₃	1182.40	1.32%	1167.00	0.00%	
	<i>GV R</i> ₄	1188.90	1.88%	1167.00	0.00%	
A60k9(60)	<i>CMM A</i>	1354.67	0.05%	1354.00	0.00%	1354
	<i>GV R</i> ₁	1377.90	1.77%	1358.00	0.30%	
	<i>GV R</i> ₂	1377.90	1.77%	1354.00	0.00%	
	<i>GV R</i> ₃	1387.90	2.50%	1358.00	0.30%	
	<i>GV R</i> ₄	1372.40	1.36%	1358.00	0.30%	
A69k9(69)	<i>CMM A</i>	1162.37	0.29%	1159.00	0.00%	1159
	<i>GV R</i> ₁	1180.60	1.86%	1159.00	0.00%	
	<i>GV R</i> ₂	1182.00	1.98%	1165.00	0.52%	
	<i>GV R</i> ₃	1180.00	1.81%	1165.00	0.52%	
	<i>GV R</i> ₄	1179.90	1.80%	1165.00	0.52%	
A80k10(80)	<i>CMM A</i>	1778.47	0.88%	1763.00	0.00%	1763
	<i>GV R</i> ₁	1811.20	2.73%	1764.00	0.06%	
	<i>GV R</i> ₂	1813.80	2.88%	1777.00	0.79%	
	<i>GV R</i> ₃	1819.30	3.19%	1781.00	1.02%	
	<i>GV R</i> ₄	1811.00	2.72%	1774.00	0.62%	
B57k7(57)	<i>CMM A</i>	1140.00	0.00%	1140.00	0.00%	1140
	<i>GV R</i> ₁	1141.10	0.10%	1140.00	0.00%	
	<i>GV R</i> ₂	1141.20	0.11%	1140.00	0.00%	
	<i>GV R</i> ₃	1141.70	0.15%	1140.00	0.00%	
	<i>GV R</i> ₄	1140.70	0.06%	1140.00	0.00%	
B63k10(63)	<i>CMM A</i>	1538.30	2.83%	1537.00	2.74%	1496
	<i>GV R</i> ₁	1546.60	3.38%	1507.00	0.74%	
	<i>GV R</i> ₂	1544.00	3.21%	1496.00	0.00%	
	<i>GV R</i> ₃	1547.00	3.41%	1516.00	1.34%	
	<i>GV R</i> ₄	1545.70	3.32%	1497.00	0.07%	
B78k10(78)	<i>CMM A</i>	1226.00	0.41%	1221.00	0.00%	1221
	<i>GV R</i> ₁	1266.00	3.69%	1233.50	1.02%	
	<i>GV R</i> ₂	1254.60	2.75%	1224.00	0.25%	
	<i>GV R</i> ₃	1249.50	2.33%	1223.00	0.16%	
	<i>GV R</i> ₄	1252.30	2.56%	1224.00	0.25%	

Table 3.2: Results on CE benchmarks (The best performing algorithm of the respective CVRP instance is highlighted in Bold)

$i(n)$	k	$\bar{C}$	$\overline{gap}$	C	gap	C_{best}
E76k7(76)	<i>CMM A</i>	685.13	0.46%	682.00	0.00%	682
	<i>GV R</i> ₁	705.90	3.50%	683.00	0.15%	
	<i>GV R</i> ₂	704.80	3.34%	687.00	0.73%	
	<i>GV R</i> ₃	703.40	3.14%	691.00	1.32%	
	<i>GV R</i> ₄	705.00	3.37%	683.00	0.15%	
E76k8(76)	<i>CMM A</i>	737.43	0.33%	735.00	0.00%	735
	<i>GV R</i> ₁	755.90	2.84%	737.00	0.27%	
	<i>GV R</i> ₂	755.30	2.76%	738.00	0.41%	
	<i>GV R</i> ₃	756.20	2.88%	738.00	0.41%	
	<i>GV R</i> ₄	755.40	2.78%	740.00	0.68%	
E76k10(76)	<i>CMM A</i>	837.53	0.91%	832.00	0.24%	830
	<i>GV R</i> ₁	851.70	2.61%	837.00	0.84%	
	<i>GV R</i> ₂	856.60	3.20%	841.00	1.33%	
	<i>GV R</i> ₃	852.60	2.72%	830.00	0.00%	
	<i>GV R</i> ₄	854.00	2.89%	837.00	0.84%	
E76k14(76)	<i>CMM A</i>	1028.10	0.70%	1021.00	0.00%	1021
	<i>GV R</i> ₁	1042.90	2.14%	1028.00	0.69%	
	<i>GV R</i> ₂	1043.50	2.20%	1022.00	0.10%	
	<i>GV R</i> ₃	1045.20	2.37%	1031.00	0.98%	
	<i>GV R</i> ₄	1043.00	2.15%	1030.00	0.88%	

Table 3.3: Results on Christofides benchmarks (The best performing algorithm of the respective CVRP instance is highlighted in Bold)

$i(n)$	k	C	gap	C_{best}
c1(50)	<i>CMM A</i>	524.61	0.00%	524.61
	<i>BB</i>	524.61	0.00%	
	<i>BR</i>	524.61	0.00%	
	<i>CGL</i>	524.61	0.00%	
	<i>GHL</i>	524.61	0.00%	
	<i>GTS</i>	524.61	0.00%	
	<i>OS</i>	524.61	0.00%	
	<i>RR</i>	524.61	0.00%	
	<i>WH</i>	524.61	0.00%	
c2(75)	<i>CMM A</i>	835.77	0.06%	835.26
	<i>BB</i>	835.26	0.00%	
	<i>BR</i>	835.26	0.00%	
	<i>CGL</i>	835.45	0.02%	
	<i>GHL</i>	835.77	0.06%	
	<i>GTS</i>	838.60	0.40%	
	<i>OS</i>	844.00	1.05%	
	<i>RR</i>	835.32	0.01%	
	<i>WH</i>	835.80	0.06%	
c3(100)	<i>CMM A</i>	826.14	0.00%	826.14
	<i>BB</i>	827.39	0.15%	
	<i>BR</i>	826.14	0.00%	
	<i>CGL</i>	829.44	0.40%	
	<i>GHL</i>	829.45	0.40%	
	<i>GTS</i>	828.56	0.29%	
	<i>OS</i>	838.00	1.44%	
	<i>RR</i>	827.53	0.17%	
	<i>WH</i>	830.70	0.55%	
c4(150)	<i>CMM A</i>	1031.63	0.31%	1028.42
	<i>BB</i>	1036.16	0.75%	
	<i>BR</i>	1030.88	0.24%	
	<i>CGL</i>	1038.44	0.97%	
	<i>GHL</i>	1036.16	0.75%	
	<i>GTS</i>	1033.21	0.47%	
	<i>OS</i>	1044.35	1.55%	
	<i>RR</i>	1044.35	1.55%	
	<i>WH</i>	1038.50	0.98%	
c5(199)	<i>CMM A</i>	1298.05	0.51%	1291.45
	<i>BB</i>	1324.06	2.53%	
	<i>BR</i>	1314.11	1.75%	
	<i>CGL</i>	1305.87	1.12%	
	<i>GHL</i>	1322.65	2.42%	
	<i>GTS</i>	1318.25	2.08%	
	<i>OS</i>	1334.55	3.34%	
	<i>RR</i>	1334.55	3.34%	
	<i>WH</i>	1321.30	2.31%	
c11(120)	<i>CMM A</i>	1042.12	0.00%	1042.11
	<i>BB</i>	1043.11	0.10%	
	<i>BR</i>	1042.11	0.00%	
	<i>CGL</i>	1074.13	3.07%	
	<i>GHL</i>	1073.47	3.01%	
	<i>GTS</i>	1042.87	0.07%	
	<i>OS</i>	1043.00	0.09%	
	<i>RR</i>	1042.11	0.00%	
	<i>WH</i>	1043.40	0.12%	
c12(100)	<i>CMM A</i>	819.56	0.00%	819.56
	<i>BB</i>	819.56	0.00%	
	<i>BR</i>	819.56	0.00%	
	<i>CGL</i>	819.56	0.00%	
	<i>GHL</i>	819.56	0.00%	
	<i>GTS</i>	819.56	0.00%	
	<i>OS</i>	819.59	0.00%	
	<i>RR</i>	819.56	0.00%	
	<i>WH</i>	819.60	0.00%	

Table 3.4: Results on Golden benchmarks [Pb1-Pb10] (The best performing algorithm of the respective CVRP instance is highlighted in Bold)

$i(n)$	k	C	gap	C_{best}
Pb1(240)	<i>CMMA</i>	5674.09	0.83%	5627.54
	<i>BR</i>	5676.97	0.88%	
	<i>GTS</i>	5736.15	1.93%	
	<i>RTR</i>	5834.60	3.68%	
	<i>SEEA</i>	5648.04	0.36%	
Pb2(320)	<i>CMMA</i>	8462.13	0.17%	8447.92
	<i>BR</i>	8512.64	0.77%	
	<i>GTS</i>	8553.03	1.24%	
	<i>RTR</i>	9002.26	6.56%	
	<i>SEEA</i>	8459.73	0.14%	
Pb3(400)	<i>CMMA</i>	11088.62	0.47%	11036.23
	<i>BR</i>	11199.72	1.48%	
	<i>GTS</i>	11402.75	3.32%	
	<i>RTR</i>	11879.95	7.65%	
	<i>SEEA</i>	11036.23	0.00%	
Pb4(480)	<i>CMMA</i>	13732.40	0.79%	13624.53
	<i>BR</i>	13637.53	0.10%	
	<i>GTS</i>	14910.62	9.44%	
	<i>RTR</i>	14639.32	7.45%	
	<i>SEEA</i>	13728.80	0.77%	
Pb5(200)	<i>CMMA</i>	6460.98	0.00%	6460.98
	<i>BR</i>	6460.98	0.00%	
	<i>GTS</i>	6697.53	3.66%	
	<i>RTR</i>	6702.73	3.74%	
	<i>SEEA</i>	6460.98	0.00%	
Pb6(280)	<i>CMMA</i>	8412.90	0.00%	8412.88
	<i>BR</i>	8429.28	0.19%	
	<i>GTS</i>	8963.32	6.54%	
	<i>RTR</i>	9016.93	7.18%	
	<i>SEEA</i>	8412.90	0.00%	
Pb7(360)	<i>CMMA</i>	10267.50	0.71%	10195.56
	<i>BR</i>	10216.50	0.21%	
	<i>GTS</i>	10547.44	3.45%	
	<i>RTR</i>	11213.31	9.98%	
	<i>SEEA</i>	10267.50	0.71%	
Pb8(440)	<i>CMMA</i>	11873.81	1.80%	11663.55
	<i>BR</i>	11936.16	2.34%	
	<i>GTS</i>	12036.24	3.20%	
	<i>RTR</i>	12514.20	7.29%	
	<i>SEEA</i>	11865.40	1.73%	
Pb9(255)	<i>CMMA</i>	586.22	0.48%	583.39
	<i>GTS</i>	593.35	1.71%	
	<i>RTR</i>	587.09	0.63%	
	<i>SEEA</i>	596.89	2.31%	
	<i>XK</i>	589.10	0.98%	
Pb10(323)	<i>CMMA</i>	744.48	0.33%	742.03
	<i>GTS</i>	751.66	1.30%	
	<i>RTR</i>	749.15	0.96%	
	<i>SEEA</i>	751.41	1.26%	
	<i>XK</i>	746.56	0.61%	

Table 3.5: Results on Golden benchmarks [Pb11-Pb20] (The best performing algorithm of the respective CVRP instance is highlighted in Bold)

$i(n)$	k	C	gap	C_{best}
Pb11(399)	<i>CMMA</i>	926.83	0.91%	918.45
	<i>GTS</i>	936.04	1.92%	
	<i>RTR</i>	934.33	1.73%	
	<i>SEEA</i>	939.74	2.32%	
	<i>XK</i>	932.68	1.55%	
Pb12(483)	<i>CMMA</i>	1125.07	1.61%	1107.19
	<i>GTS</i>	1147.14	3.61%	
	<i>RTR</i>	1137.18	2.71%	
	<i>SEEA</i>	1152.88	4.13%	
	<i>XK</i>	1140.72	3.03%	
Pb13(252)	<i>CMMA</i>	869.23	1.18%	859.11
	<i>GTS</i>	868.80	1.13%	
	<i>RTR</i>	881.04	2.55%	
	<i>SEEA</i>	877.71	2.17%	
	<i>XK</i>	881.07	2.56%	
Pb14(320)	<i>CMMA</i>	1086.74	0.50%	1081.31
	<i>GTS</i>	1096.18	1.38%	
	<i>RTR</i>	1103.69	2.07%	
	<i>SEEA</i>	1089.93	0.80%	
	<i>XK</i>	1118.09	3.40%	
Pb15(396)	<i>CMMA</i>	1361.20	1.19%	1345.23
	<i>GTS</i>	1369.44	1.80%	
	<i>RTR</i>	1364.23	1.41%	
	<i>SEEA</i>	1371.61	1.96%	
	<i>XK</i>	1377.79	2.42%	
Pb16(480)	<i>CMMA</i>	1646.63	1.48%	1622.69
	<i>GTS</i>	1652.32	1.83%	
	<i>RTR</i>	1657.93	2.17%	
	<i>SEEA</i>	1650.94	1.74%	
	<i>XK</i>	1656.66	2.09%	
Pb17(240)	<i>CMMA</i>	710.16	0.34%	707.79
	<i>GTS</i>	711.07	0.46%	
	<i>RTR</i>	720.44	1.79%	
	<i>SEEA</i>	717.09	1.31%	
	<i>XK</i>	747.24	5.57%	
Pb18(300)	<i>CMMA</i>	1007.90	0.92%	998.73
	<i>GTS</i>	1016.83	1.81%	
	<i>RTR</i>	1029.21	3.05%	
	<i>SEEA</i>	1018.74	2.00%	
	<i>XK</i>	1066.57	6.79%	
Pb19(360)	<i>CMMA</i>	1384.82	1.31%	1366.86
	<i>GTS</i>	1400.96	2.49%	
	<i>RTR</i>	1403.05	2.65%	
	<i>SEEA</i>	1385.60	1.37%	
	<i>XK</i>	1435.88	5.05%	
Pb20(420)	<i>CMMA</i>	1862.09	2.25%	1821.15
	<i>GTS</i>	1915.83	5.20%	
	<i>RTR</i>	1875.17	2.97%	
	<i>SEEA</i>	1846.55	1.39%	
	<i>XK</i>	1934.99	6.25%	

- C - denotes the best fitness among the converged solutions on a VRP instance
- $\overline{C}$ - denotes the average fitness of the converged solutions on a VRP instance
- C_{best} - denotes the best known result of a VRP instance reported in the literature
- gap - denotes the minimum percentage fitness deviation from C_{best} as given by $\frac{C - C_{best}}{C_{best}} \times 100\%$
- $\overline{gap}$ - denotes the average percentage fitness deviation from C_{best} as given by $\frac{\overline{C} - C_{best}}{C_{best}} \times 100\%$
- t - denotes the computation time, in seconds, to arrive at the converged solution
- $\bar{t}$ - denotes the average computation time, in seconds, to arrive at the converged solutions
- g - denotes the average number of generations elapsed before the solution converged

parent selection scheme. The CMMA is pitted against other state-of-the-art stochastic search algorithms that have reported competitive or new best known results on the representative VRP benchmarks included here. besides the CMMA in Tables 3.1, 3.2, 3.4, 3.5, and 3.3 are the reported published results of state-of-the-art algorithms used in the present comparison study on these algorithms. At the same time, the plots of minimum and average fitness deviation from best known results as depicted in Figs. 3.6, 3.7, 3.9, 3.8.a and 3.8.b highlighted the superior search quality of CMMA over all the stochastic search algorithms considered (except for instance B63K10(63) in Figs. 3.6.a) on minimum gap, in terms of both the best and average fitness of the converge solutions obtained. On the Golden benchmark set, which includes the large-scale CVRP instances with problem size 240-483, CMMA also showcased results which are competitive or superior to the SEEA and BR state-of-the-art memetic algorithms.

 3.6.a: Plot of minimum *gap* across CVRP instances

 3.6.b: Plot of average *gap* across CVRP instances

Figure 3.6: Plot of minimum and average percentage fitness deviations from the best known results across Augerat benchmarks

 3.7.a: Plot of minimum *gap* across CVRP instances

 3.7.b: Plot of average *gap* across CVRP instances

Figure 3.7: Plot of minimum and average percentage fitness deviations from the best known results across CE benchmarks

 3.8.a: minimum *gap* over Golden benchmarks (Prob 1-10)

 3.8.b: minimum *gap* over Golden benchmarks (Prob 11-20)

Figure 3.8: Plot of minimum percentage fitness deviation from the best known results across Golden benchmarks

Figure 3.9: Plot of minimum percentage fitness deviation from the best known results across Christofides benchmarks

As noted in the literature review for CVRP, state-of-the-art memetic algorithms for CVRP, including SEEA [170], BR [209], Active Guided Evolution Strategy (AGES) [121], and the D-Ants algorithm [175], are considered to be extremely effective algorithms for solving the problem. However, they require large and complex domain knowledge incorporation specifically for VRP, thus raising the implementation efforts necessary. On the other hand, the proposed CMMA remains a simple hybrid that is equipped with a *multi-memes* individual learning capable of synergizing simple local search heuristics or memes into more effective solver that produces competitive or superior search performances over humanly crafted domain-specific algorithms like BR and SEEA. Taking inspiration from Occam's razor for selection among competing hypotheses, it can be stated that among the competing algorithms for CVRP, one that makes fewest implementation twists in terms of domain knowledge incorporation, in this case CMMA, would be preferred. Particularly, the simpler the algorithm, the better it is!

The efficiency of the proposed CMMA is demonstrated in Subsection 4.5.2, where the convergence of CMMA is shown to be around 0.1 second for Augerat, CE benchmark sets, around

1 second for the Christofides benchmark set, and around 1 minute for the large scale Golden benchmark set, with computational cost lower than adaptive memetic algorithms proposed by other authors such as SGMA.

3.6 Conclusions

In this chapter, a simple hybrid CMMA with a multi-memes individual learning is introduced that combines multiple distinct memes during search, each having different problem-specific learning roles and search features. Our motivation is to design a simple hybrid with multi-memes individual learning which overcomes the inherent risk associated with choosing a single meme to be used for individual learning, as that meme carries search bias whose performance may vary greatly from one CVRP class to another, as well as through different population-based search stage g . The multi-memes individual learning reduces premature search stagnation and increases convergence reliability, by hybridizing multiple memes in a *cooperative* manner to allow these memes working together to accomplish the shared optimization goal. CMMA is relatively simple to implement and the empirical study highlighted its high effectiveness and efficiency.

Chapter 4

A Conceptual Modeling of Meme Complexes in Adaptive Hybrids

While the multi-memes individual learning for simple hybrids proves to be effective and efficient in solving CVRPs, the success of multi-memes individual learning and hence simple hybrids in general, greatly depends very much on the human expert who needs to be equipped with detailed knowledge of the problem domain. For example, CMMA in Chapter 3 designed for solving CVRPs includes the three CVRP-specific individual learning stages of clustering → merging → TSP routing optimization. In this chapter, the ideas of multi-memes introduced in Chapter 3 are further explored and evolved into the conceptual modeling of co-adapted meme complexes or *memeplexes* for adaptive hybrids as a self-configuring search methodology that is capable of learning and adapting the multi-memes individual learning to the given problem of interests while the search progresses online. The *Co-Adapted Memetic Algorithm* (CAMA) which implements the adaptive memeplex individual learning is then proposed and studied in the context of solving CVRPs.

4.1 Introduction

Adaptive hybrids in memetic computation deploy evolutionary operators (such as crossover and mutation) to generate one or more solutions, each of which is then subject to local refinement

by a meme (adaptively selected or generated) in subsequent individual learning. With respect to individual learning, the workflow of a canonical adaptive hybrid is illustrated in Alg. 7.

Algorithm 7 Outline of Canonical Adaptive Hybrid

Input: Problem Instance

Output: Global best solution s_g

Create an initial population of solutions $\text{Pop}(g = 0) = \{s_1, s_2, \dots, s_i, \dots, s_M\}$

while stopping conditions are not satisfied **do**

$g \leftarrow g + 1$

Perform evolutionary operators to generate $\text{Pop}(g)$ from $\text{Pop}(g - 1)$ based on $f(s_i)$

Evaluate $f(s_i), \forall s_i \in \text{Pop}(g)$

 /* Outline of adaptive individual learning */

for $s_i, \forall s_i \in \text{Pop}(g)$ **do**

 /* Adaptive solution selection for individual learning */

if $\text{IsSelectedForIndividualLearning}(s_i)$ **then**

$t \leftarrow 0$

 /* Adaptive meme selection or generation for individual learning */

$m \leftarrow \text{MemeSelection}(s_i, \text{Pop}(g))$

 /* Adaptive search intensity */

while $\text{IsIndividualLearningTerminated}(s_i, t, m)$ **do**

$s_i \leftarrow \text{Refine}(s_i, m)$

$t \leftarrow t + 1$

end while

$\text{MemeEvaluation}(\Delta f(s_i)_m, \Delta T_m, m)$

end if

end for

Update global best solution s_g

end while

As illustrated in Alg. 7, an adaptive hybrid in the form of adaptive memetic algorithm starts with the initialization of a population $\text{Pop}(g = 0)$ of candidate solutions. At each generation g , a population of offspring $\text{Pop}(g) = \{s_1, s_2, \dots, s_M\}$ is generated using evolutionary search through reproduction operators such as crossover and mutation. The $\text{Pop}(g)$ subsequently undergoes fitness evaluation and adaptive individual learning, and update the global best solution.

This workflow of memetic search proceeds until some stopping conditions are reached. As illustrated in Alg. 7, meme adaptation can be typically applied to:

- Adaptive Selection of solutions for Individual learning: adaptively select a subset of individuals from $\text{Pop}(g)$ to undergo individual learning, as in the method *IsSelectedForIndividualLearning()*.
- Adaptive Meme Selection: adaptively select or generates a meme for a particular solution, as in the method *MemeSelection()*.
- Adaptive Search Intensity: adaptively adjust the search intensity based on the current solution being refined, as in the method *IsIndividualLearningTerminated()*.
- Meme Evaluation: reconfigure the meme selection or generation controller based on feedback from the individual learning, $\Delta f(s)_m, \Delta T_m$ are the fitness gain and computational cost associated with using m to refine s , respectively, as in the method *MemeEvaluation()*.

For hybrids in memetic computation, the adaptive selection of memes during the stochastic search is not a new concept. As highlighted in Chapter 2, many meme adaptation approaches have been proposed in memetic computation [149], such as *meta-Lamarckian learning* [147], *simple inheritance mechanism* [97], *cost-benefit-based adaptation scheme* [80], *self-generating MAs* [95]. The meme selection can be based on feedback such as population diversity [135, 206], or other search performance metrics obtained from the individual learning process [136, 79], or the meme can be adaptively self-assembled such as in [95, 194]. The adaptive algorithm presented in Alg. 7 degenerates into hyper-heuristic [157] when the size of $\text{Pop}(g)$ is reduced to 1 and the stochastic reproduction operators (such as evolutionary operators) are removed.

It is worth noting that many of these adaptive hybrids are based on single-meme individual learning in which only a single meme is selected or built for each solution during local

refinement. On the other hand, the meme co-adaptation for the multi-memes individual learning, in which multiple memes adaptively combine with each other to form cohesive emerging memeplexes for refining each solution on the problem domain being solved, has not yet been explored.

There are benefits on memeplex adaptation over single-meme adaptation for individual learning in adaptive hybrids. Single-meme adaptive individual learning can suffer from a common limitation known as a two-timescale problem. The amount of time needed to efficiently find those fit genetic solutions that are easy to locate with a given individual learning operator can be a great deal less than that needed to find the fittest meme in the first place. For some problems this may not be the case. On the other hand, the adaptive memeplex individual learning operators are being used to find good combination of memes for the solutions of a problem instead of the fittest meme. Adaptive combination of differing memes within memeplex individual learning for the memetic search reduces the risks of failing on problems of different types as well as making discovery of high-quality solutions more likely, by investing the local search time on multiple memes and their synergy, instead of betting the entire local search time budget on a single meme during the local refinement.

To induce the desired meme co-adaptation of multi-memes individual learning, this dissertation introduces a structural form of memetic existence, i.e., co-adapted meme complexes or memeplexes, in the individual learning phase of adaptive hybrids. A memeplex is essentially a co-adapted stable set of mutually assisting memes [40], working together to achieve more than what each meme could accomplish on its own [192, 16]. In this chapter, we focus on the concept of *memeplex* as a self-organizing paradigm for problem-solving whereby memes engage in collective learning and actions that eventually bring higher efficacy to stochastic search. Represented in the neural wiring structure or so-called memetic network, a memeplex takes the form of a distributed and dynamically evolving network with memes complementing one another on the search. Each meme can assume a functional status from the network's

spatial conformation and induces a unique learning bias and search neighborhood structure. The emerging memeplex has the capability of reconfiguring itself to adapt to the domain or instances of problems being solved and acclimatize to suit a given problem in hand by methodically utilizing acquired information about the matching of problem domain to the stochastic search dynamics.

The rest of the chapter is organized as follows. Section 4.2 begins with the introduction of memeplexes and their proposed neural wiring structure in Subsections 4.2.1 and 4.2.2. Subsequently, Section 4.3 discusses the design of memeplex individual learning, including suitable credit assignment criteria for facilitating the emergence of highly cooperative memes in the memeplex for individual learning as well as the memeplex generation scheme. The *Co-Adapted Memetic Algorithm (CAMA)*, which takes the proposed conceptual modeling of memeplexes for solving CVRPs is then proposed in Section 4.4. This is followed by the empirical assessment of CAMA against several existing state-of-the-art adaptive or human-designed memetic algorithms and metaheuristics, on the set of CVRPs. Empirical results obtained showed that co-adapted meme complexes provide a means of creating highly robust and scalable algorithms, by generating improved or at least competitive search performances over existing state-of-the-art approaches. Finally, the last section ends with a brief conclusion of the present research.

4.2 Memeplex in Memetic Computation

Scientists in the field of memetics have argued that memes do not work alone, but bundle themselves in so-called meme complexes or memeplexes, so as to work more effectively [16, 192, 40]. In science, a memeplex represents an ideology or belief system, e.g. religion, that exist in the mental state of an individual during its lifetime. Structurally, a memeplex is a co-adapted stable set of mutually assisting memes [40], and can arise in the meme pool of seemingly unconnected memes [16]. Within a memeplex, memes interact to reinforce each

other, forming a structure of memetic stability, with each meme being supported and promoted by the other memes in the memplex.

In stochastic search of adaptive hybrids, the concept of a memplex creates a self-consistent paradigm whereby memes self-organize and engage in collective learning and action that will eventually give higher efficacy to individual learning process. In a search algorithm, memplexes are essentially sets of instructions or memes that carry out the search behaviors or moves in the individual learning. Having several memes which co-adapt with each other, the memplex forges a diverse and integrated memetic ecosystem because each meme can take on a specific role such as inducing a unique learning bias and search neighborhood structure, with memes complementing one another on the search. This section presents our representation of memplexes in search, the credit assignment criteria for evaluating individual memes and memes co-adaptation, and the role of emergent memplexes in the individual learning process of an adaptive hybrid.

4.2.1 Memplex Representation

In our study, a memplex is perceived in the form of a neural wiring structure housed within the agent's brain that controls the agent's behavior in the individual learning lifetime, as illustrated in Fig. 4.1.

Figure 4.1: Memeplex in its neural wiring structure

Symbolically, a memeplex candidate $\mathcal{M}_\pi$ associated with solution s_i , can be built from the meme terminal set. The representation of a memeplex $\mathcal{M}$ is defined as a sequence of memes, as shown below:

$$\mathcal{M}_\pi = m_{\pi(1)} \rightarrow m_{\pi(2)} \rightarrow \dots \rightarrow m_{\pi(k)} \rightarrow \dots \rightarrow m_{\pi(\varkappa)}$$

Where $\pi = \{i | i \in \mathbb{N}\}$ is the memeplex policy for $\mathcal{M}_\pi$, and $\varkappa = |\pi|$ denotes the size of the memeplex $\mathcal{M}_\pi$. If $\pi = \{2, 4, 1, 2\}$, then $\varkappa = 4$, and $\mathcal{M}_\pi = m_2 \rightarrow m_4 \rightarrow m_1 \rightarrow m_2$. Subsequently, the memeplex candidate $\mathcal{M}_\pi$ is sometimes abbreviated as $\mathcal{M}$ in the text for convenience.

The memes that form a memeplex $\mathcal{M}$ are taken from the meme terminal set Ψ

$$\Psi = \{m_1, m_2, \dots, m_\kappa\}$$

Each meme $m_k \in \Psi$ encodes the unit of instruction for performing local refinement during the individual learning process. In general, a meme m_k is defined by the meme tuple (refers to Section 3.2 Chapter 3), as shown below:

$$m_k = (\langle T_i \rangle, \langle Acceptance_Strategy \rangle, \langle Search_Budget \rangle)$$

where

- $\langle T_i \rangle_{m_k}$ denotes the move operator of the search neighborhood (e.g. the edge-exchange neighborhood in Chapter 3)
- $\langle Acceptance_Strategy \rangle_{m_k}$ denotes the acceptance strategy of the local refinement associated with m_k
- $\langle Search_Budget \rangle_{m_k}$ denotes the search intensity of m_k during local refinement

During the individual learning process, a memplex candidate $\mathcal{M}$ refines a solution s by iteratively mobilizing the memes it encodes to conduct local refinement on s . The activation of meme $m_i \in \mathcal{M}$ at any given time instance is thus tightly coupled with the preceding mobilized meme $m_j \in \mathcal{M}$, which has a connectivity weight denoted by w_{ij} . In $\mathcal{M}$, each two memes $m_i, m_j \in \mathcal{M}$ are wired to each other with some connectivity weights w_{ij} and w_{ji} , which forms a connection topology network¹. The neural wiring structure of $\mathcal{M}$, a.k.a, memetic network of $\mathcal{M}$ is illustrated in Fig. 4.1.

The memetic network topology of $\mathcal{M}$ can thus be represented as a fully-connected directed graph, with the set of memes as the vertex set, and a set of directed edges $\{\vec{e}_{ij}\}$ indicating that the associations or synergies between meme m_j and m_i . The memetic network can be represented using a matrix $W_{\mathcal{M}}$, given by:

¹In terms of a neural network, the connection topology is a specification of which neurons are connected to which other neurons, in the form of a directed graph

$$W_{\mathcal{M}} = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1\kappa} \\ w_{21} & w_{22} & \cdots & w_{2\kappa} \\ \vdots & \vdots & \ddots & \vdots \\ w_{\kappa 1} & w_{\kappa 2} & \cdots & w_{\kappa \kappa} \end{pmatrix} \quad (\text{Eq. 4.1})$$

where w_{ij} refers to the connectivity (or synergy) from meme m_i to m_j , and can be perceived as a preference value that defines the degree to which m_i 's activation favors the subsequent activation of m_j . w_{ii} denotes the individual fitness level of meme m_i . The network topology of $\mathcal{M}$ as modeled by $W_{\mathcal{M}}$ is essentially a dynamic and distributed complex network woven in memetic connection patterns. To simplify the symbol w_{ii} , we denote it as w_i in the writing. Fig. 4.2 illustrates the network topology that shows parts of the connectivity.

Figure 4.2: The network topology of memplex candidate displaying part of connectivities

4.2.2 Meme Co-adaptation

The memplex representation allows memes within the memetic network topology to draw their functional status from their spatial arrangement, by interacting together to achieve advanced individual learning capabilities. The non-linear memetic interaction within forges a

neural wiring pattern that produces unique combined effects during the individual learning process. The interaction of one meme's activation with that of its immediate surrounding meme in the memetic network triggers a complex chain of responses leading to some higher-order behavior. As a result, the emergent memeplex is unlike its component memes insofar as it has a distinctive effect on individual learning that cannot be reduced to their sum or their difference [111, 17]. The number of interactions between memes in a memetic network increases combinatorially with the number of memes, thus potentially allowing for many new and subtle types of behavior to emerge.

The arising novel and coherent structures, patterns and properties of the emergent memeplexes must be co-determined by the problem context and the individual learning interactions between memeplexes [64]. Further, the emergence of memeplexes presupposes that the recurrent and persistent patterns of memetic activity can arise out of a multiplicity of relatively simple memetic interactions. As a process designed to fulfil these requirements, memetic evolution provides us with a cognitive model for the growth of powerful memeplexes. During the memetic evolution, memes are filtered out of the meme pool and get connected somehow, based on the selection pressure established in the problem context and the memeplex patterns of accumulating change observed from the causal interaction between memes in the individual learning. In this process, the matrix $W_{\mathcal{M}}$ becomes the vehicle for memeplex propagations, turning the memetic network into a distributed and dynamically evolving network capable of adapting to the problem context. The emergence of intelligent memeplexes are commonly identified by the resultant memetic stability uncovered for the co-adapted memes [179]. This is in turn determined by the fitness of the elementary memes and the synergies among them in the stabilized memetic network. In the next section, these processes and properties are discussed in greater details.

4.3 Role of Memeplex in Individual Learning

The role of a memeplex candidate $\mathcal{M}_\pi$ in individual learning is to serve as a recipe that intelligently combines appropriate memes within the memetic network for the refinement of each solution s produced by the stochastic reproduction operators in the global search. The outline of adaptive memeplex individual learning is illustrated in Alg. 8.

Algorithm 8 Outline of Memeplex-Individual-Learning

Input: s (solution to refine)

Output: s' (solution refined)

$s \leftarrow s'$

$t \leftarrow 0$

/ Adaptive memeplex generation */*

$\mathcal{M}_\pi = \{m_{\pi(k)}\}_{k=1}^{k=\varkappa} \leftarrow \text{AssembleMemeplexCandidate}(\Psi, W_{\mathcal{M}})$

/ Memeplex solution refinement */*

for $k = 1 \dots \varkappa$ **do**

/ Adaptive search intensity */*

while $\text{IsMemeplexIndividualLearningTerminated}(s, t, m_{\pi(k)})$ **do**

$s' \leftarrow \text{Refine}(s', m_{\pi(k)})$

/ Credit Assignment to individual memes */*

$w_{\pi(k)} \leftarrow \text{EvaluateMeme}(w_{\pi(k)}, \Delta f(s')_{m_{\pi(k)}}, \Delta T_{m_{\pi(k)}})$

$t \leftarrow t + 1$

end while

end for

/ Credit Assignment to co-adapted memes */*

$W_{\mathcal{M}} \leftarrow \text{EvaluateMemeplex}(W_{\mathcal{M}}, \Delta f(s')_{\mathcal{M}_\pi}, \Delta T_{\mathcal{M}_\pi})$

Return s'

As illustrated in Alg. 8, the activity of a memeplex candidate $\mathcal{M}_\pi$ during an individual learning process, a.k.a. $\mathcal{M}_\pi$'s life cycle, is a series of $\varkappa$ -stage memetic activations $m_{\pi(1)} \rightarrow m_{\pi(2)} \rightarrow \dots \rightarrow m_{\pi(\varkappa)}$ on a solution s , with each individual learning stage k mobilizing a

meme $m_{\pi(k)} \in \mathcal{M}_\pi$ to refine solution s . When activated at the k^{th} stage of the individual learning, $m_{\pi(k)}$ induces its learning bias and search neighborhood structure, as defined by its move operator $\langle T_i \rangle_{m_{\pi(k)}}$ and move acceptance criteria $\langle Acceptance_Criteria \rangle_{m_{\pi(k)}}$, to refine solution s .

During the individual learning process, search performance of the memplex as well as each of its activated memes is recorded and used to adjust each individual meme's fitness as well as the memetic synergy between memes. This processed feedback is then assimilated in credit assignment to individual meme fitness and memetic synergy via connectivity adjustment to $W_{\mathcal{M}}$ (as in the *EvaluateMeme* and *EvaluateMemplex* procedures in Alg. 8), thus allowing individual meme fitness and memetic synergy to synchronize with the changing state of the search. The detailed steps to synchronize the individual meme fitness and memetic synergy as well as the memplex network adaptation with the changing state of a search, are described in the next section.

To grasp the major differences between established meme adaptation approaches in literature and meme co-adaptation in memplex individual learning proposed here, please refer to Fig. 4.3 which compares and contrasts the schematic workflow of both.

Figure (a) Workflow of established Meme Adaptation

Figure (b) Workflow of Memeplex Individual Learning

Figure 4.3: Workflow Comparison of Individual-Learning by Meme Adaptation and Memeplex

As illustrated in Fig. 4.3-(a), for established meme adaptation techniques in hybrids, a meme $m \in \Psi$ is adaptively selected or generated for each solution $s \in \text{Pop}(g)$ at the beginning of its individual learning process. This meme m is then assigned to refine s until it reaches the pre-determined or auto-configured search depth. Credit assignment or other method is subsequently applied to evaluate m based on its recent search performance. The individual learning phase continues with another solution $s' \in \text{Pop}(g)$ until all selected individuals have

undergone local refinement and the next population reproduction phase starts. It is worth noting that while a different meme may be assigned for refining each solution $s \in \text{Pop}(g)$, the solution refinement search of s is constrained to only one single neighborhood structure as defined by the assigned meme's move operator and acceptance criteria. In other words, throughout the whole individual learning period, the entire local search time budget for a solution s is allocated to a single meme m assigned to refine the solution s . In light of the simple hybrid ideas presented in Chapter 3, established meme adaptation techniques are thus very much akin to an adaptive *single-meme* individual learning.

In contrast to established meme adaptation techniques, the adaptive memeplex individual learning introduced here involves an adaptive multi-meme selection scheme in the individual learning for multiple memes to be assigned for refining a solution $s \in \text{Pop}(g)$. For example, as illustrated in Fig. 4.3-(b), the memeplex candidate $\mathcal{M}_\pi$ combines three different memes m_2, m_3, m_4 to refine solution $s_1 \in \text{Pop}(g)$. In other words, the solution refinement search of s_1 can now move in three different neighborhood structures defined by these memes. This reduces the risk of suffering from premature local search convergence by investing the local search time budget in multiple memes, since the chance of all the constituent memes being stuck at the same point in the solution space is quite small. The memetic representation of memeplex, when combined with meme co-adaptation mechanisms, allows intelligent memeplexes to emerge which have following properties that acclimatize towards matching the problem being solved:

- ☞ Adaptively adjust the search intensity of the individual learning by controlling the size of a memeplex candidate, $\varkappa$, as well as adaptively selecting memes with varied search budgets into the memeplex candidate. For example, in Fig. 4.3-(b), memeplex candidate $\mathcal{M}' = m_1 \rightarrow m_4$ assigned to solution s_M has a lower search intensity compared to memeplex candidate $\mathcal{M} = m_2 \rightarrow m_4 \rightarrow m_3 \rightarrow m_4$.

- ☞ Adaptively manage the local search time budget shared among memes combined to refine a solution. For example, for memplex $\mathcal{M} = m_2 \rightarrow m_4 \rightarrow m_3 \rightarrow m_4$ illustrated in Fig. 4.3-(b), the local search time budget assigned to m_4 is more than that to m_2 and m_3 .
- ☞ Adaptively harness the unique domain-driven search biases of different memes during individual learning, by having multiple memes hybridized and combined in a *cooperative* manner (e.g., in a particular memetic activation order $m_2 \rightarrow m_4 \rightarrow m_3 \rightarrow m_4$ of $\mathcal{M}_\pi$ illustrated in Fig. 4.3-(b)).

These properties arise implicitly from meme co-adaptation based on evaluative feedback from memplex solution refinement, as memes are learnt by interaction and rewards are basically given directly by the genetic population environment after the memes are mobilized for the solution improvement. In the evaluative feedback, the reward received after each meme action gives some information about how good the action was. Such evaluative feedback depends entirely on the meme action taken throughout the memplex solution refinement process, and is the basis of credit assignment which dynamically evaluates the functional potential of each meme and their patterns of interaction and combination. Via evaluative feedback and credit assignment, memplex system learns to *control* its environment which is the genetic population $\text{Pop}(g)$, because it can now influence the genetic information it receives via its guided meme mobilization sequence. The following subsections detail the credit assignment based on evaluative feedback and the meme co-adaptation mechanisms, that facilitate the emergence of intelligent memplexes for adaptive individual learning.

4.3.1 Credit Assignment for Individual Memes

In individual learning iteration, the two processes *Refine* and *EvaluateMeme* alternate, each completing before the other begins, as illustrated in Alg. 8. When a meme gets mobilized for solution improvement in the *Refine* process, the *EvaluateMeme* process monitors, interprets and

assimilates the harvested performance information from each instance of meme mobilization that led to the attained performance, which is subsequently used to evaluate the functional potential or fitness of a meme. Specifically, each time after the *Refine* procedure activates meme $m_i \in \mathcal{M}_\pi$ on improve solution s , the *EvaluateMeme* procedure follows to compute an immediate reward r_i from the mobilization and the meme fitness w_i is re-evaluated (as in [147]), which is shown below:

$$w_i = \gamma w_i + r_i \quad (\text{Eq. 4.2})$$

$$= \gamma w_i + \frac{\Delta f(s)_{m_i}}{\Delta T_{m_i}} \quad (\text{Eq. 4.3})$$

where $\Delta f(s)_{m_i}$, ΔT_{m_i} denote the solution fitness differential gain and computational cost associated with the functional activity performed by m_i , and $0 < \gamma < 1$ is the discount factor of w_i . In Eq. 4.2, the discount factor γ reflects the influence of a meme's recent performance.

From Eq. 4.2, w_i can be thought of as a prediction of rewards for a meme m_i , and serves as a *value function* that specifies the total amount of rewards m_i can expect to accumulate over the search. Whereas reward r_i determines the immediate, intrinsic desirability of m_i , w_i indicates the *long-term* desirability of m_i after taking into accounts the other memes that are likely to follow, and the rewards resulted from their mobilizations.

4.3.2 Credit Assignment for Co-adapted Memes

It is important to note that during solution refinement by memplex candidate $\mathcal{M}_\pi$ in Alg. 8, individual learning actions by meme $m_{\pi(k)} \in \mathcal{M}_\pi$ may affect not only the immediate reward of $m_{\pi(k)}$ but also the next situation in which $m_{\pi(k+1)}, m_{\pi(k+2)} \dots$, are applied, and through that, all subsequent rewards they may receive. This information can be captured and stored as the memetic synergy between two memes. The memetic synergy is defined between any two memes m_i and m_j , and represented by w_{ij} and w_{ji} , where w_{ij} denotes the degree to which m_i 's

mobilization favors m_j 's activation. The modeling of memetic synergy is crucial since emergent order cannot arise if the coexisting memes of a memeplex do not interact [19]. Memetic synergy measures the preferences of one type of interaction over another. Through memetic synergy, memes become 'acquainted' with each other and socialize in the interaction space, thus clarifying the role of and lines of communications between memes and their integration in a memetic network topology that is optimally harnessed and coordinated. In other words, the functional potential of co-adapted memes can be evaluated by learning memetic synergies.

To learn the memetic synergy w_{ij} , the *EvaluateMemeplex* procedure is executed that assigns credits to the co-adapted memes in memeplex candidate $\mathcal{M}_\pi$ after $\mathcal{M}_\pi$ completes refining a solution s at the end of an individual learning process. In *EvaluateMemeplex*, $\mathcal{M}_\pi$ is first decoded by retracing the memetic activation profile " $\dots \rightarrow m_{\pi(k)} \rightarrow m_{\pi(k+1)} \rightarrow \dots$ " from $\mathcal{M}$'s representation and network topology. Subsequently the memetic synergy between pairs of memes $(m_{\pi(k)}, m_{\pi(k+1)})$ is updated [89], as given by:

$$w_{\pi(k)\pi(k+1)} = w_{\pi(k)\pi(k+1)} + \frac{R_{\mathcal{M}}}{\varkappa}, k = 1, \dots, \varkappa - 1 \quad (\text{Eq. 4.4})$$

$$= w_{\pi(k)\pi(k+1)} + \frac{\Delta f(s)_{\mathcal{M}}}{\varkappa * \Delta T_{\mathcal{M}}} \quad (\text{Eq. 4.5})$$

where $R_{\mathcal{M}} = \frac{\Delta f(s)_{\mathcal{M}}}{\Delta T_{\mathcal{M}}}$ denotes the delayed reward to all the memes in $\mathcal{M}_\pi$ obtained after $\mathcal{M}_\pi$ completes its local refinement on the solution. $\Delta f(s)_{\mathcal{M}}$, $\Delta T_{\mathcal{M}}$ are the overall fitness differential gain and computational cost of $\mathcal{M}$, respectively. $\Delta f(s)_{\mathcal{M}}$ is measured by the fitness improvement before and after the memeplex local refinement on the solution. $\Delta T_{\mathcal{M}}$ is measured by the number of local refinement steps made by $\mathcal{M}_\pi$ to reach the improved solution. Fig. 4.4 depicts the credit assignment to the individual meme fitness and memetic synergy as $\mathcal{M}_\pi$ undergoes a series of memetic activation during the solution refinement of s .

As illustrated in Fig. 4.4, during meme activation, immediate rewards gives better and better estimate over the meme's individual fitness while the rewards for their co-adaptation

Figure 4.4: The credit assignment for co-adapted memes as $\mathcal{M}_\pi$ refines a solution

helps identify collaborative patterns among memes which tends to improve genetic solutions. The life cycle of a memeplex candidate $\mathcal{M}$ is a multiplicity of simple interactions between the connectivity adjustment for credit assignment processes described by Eq. 4.2 and Eq. 4.4, and the non-linear interaction process of memetic activation. This memetic interaction energetics that permeate memetic network in individual learning for search is driven by the dynamics of the connectivity adjustment within $W_{\mathcal{M}}$ as triggered by memetic activation and propagating along the directed edges of the memetic network. The result of this process interplay is that stable patterns of memetic connections within $W_{\mathcal{M}}$ begin to emerge, allowing the gradual evolution of compatible memes which self-organize into memetic network topologies that exhibit memetic stabilities. Thus, the discovery of memetic network topologies in the form of recurrent and persistent neural wiring patterns of memeplexes give rise to new memetic networks that is synchronized with the changing state of the search.

4.3.3 Emergence of Memeplexes

The individual meme fitness and memetic synergy denote a form of knowledge acquired during the life cycles of previous instances of memeplex candidates, which is subsequently employed to adaptively generate new memeplex candidate for the future state of search. When a generated memeplex candidate $\mathcal{M}$ delivers great improvement on a solution, connectivity adjustments within $W_{\mathcal{M}}$ lead to reinforcement in pattern of memetic interaction. On the contrary, when $\mathcal{M}_{\pi}$ does not produce desired behavior, implying that the memes that make up $\mathcal{M}$ are non-favorable, connectivity adjustments within $W_{\mathcal{M}}$ then lead to the dissolution of $\mathcal{M}$. This process stretches across generations of memeplexes as well as populations of stochastic solutions, and efficiently converts the revealed potential of individual memes and memetic synergies, into an increasingly coherent memeplex of efficacy that is properly aligned with the structure of the solution population throughout the changing state of search. Alg. 9 illustrates the strategy to generate a memeplex candidate $\mathcal{M}$.

Algorithm 9 Outline of AssembleMemeplexCandate($W_{\mathcal{M}}, \Psi$)

Input: $W_{\mathcal{M}}, \Psi$

Output: $\mathcal{M}_{\pi}$

$M_{\pi} \leftarrow \emptyset$

$m_{\pi(1)} \leftarrow$ Select a seed meme $m_{\pi(1)} \in \Psi$ with probability $P_{seed}(m_{\pi(1)})$

$\mathcal{M}_{\pi} \leftarrow$ Add $m_{\pi(1)}$ to $\mathcal{M}_{\pi}$

$k \leftarrow 2$

while TRUE do

 Select meme $m_{\pi(k)} \in \Psi$ with probability $P_{join}(m_{\pi(k)})$

if accepted($m_{\pi(k)}$) **then**

$\mathcal{M}_{\pi} \leftarrow \mathcal{M}_{\pi} \cup m_{\pi(k)}$

$k \leftarrow k + 1$

else

Break

end if

end while

Return $\mathcal{M}_{\pi} = m_{\pi(1)} \rightarrow \dots \rightarrow m_{\pi(k)} \rightarrow \dots \rightarrow m_{\pi(\infty)}$

The adaptive memplex generation process is described in what follows. First, a ‘seed’ meme $m_{\pi(1)}$ is filtered out of the meme pool, with probability $P_{seed}(m_{\pi(1)})$ as given by:

$$P_{seed}(m_{\pi(1)}) = \frac{w_{\pi(1)}}{\sum_{\mu=1}^{\mu=\kappa} w_{\mu}} \quad (\text{Eq. 4.6})$$

where w_{μ} denotes the fitness of meme m_{μ} , and κ denotes the size of meme terminal set Ψ . From Eq. 4.6, it is worth noting that memes with higher fitness are given higher probability of being chosen as the seed. Once selected, the seed meme $m_{\pi(1)}$ will then ‘germinate’ and grow into a memplex candidate $\mathcal{M}_{\pi}$, that harnesses and integrates a cognitive territory glued together based on individual meme fitness and memetic synergy at the current search state. This cognitive territory is defined such that any subsequent meme m_k to be included in $\mathcal{M}$, is chosen from Ψ , with probability $P_{join}(m_{\pi(k)})$ defined by:

$$P_{join}(m_{\pi(k)}) = \frac{[w_{\pi(k)}]^{\alpha} [w_{\pi(k-1)\pi(k)}]^{\beta}}{\sum_{\mu=1}^{\mu=\kappa} [w_{\mu}]^{\alpha} [w_{\pi(k-1)\mu}]^{\beta}} \quad (\text{Eq. 4.7})$$

where w_{μ} denotes the fitness of meme m_{μ} , and $w_{\pi(k-1)\mu}$ denotes the memetic connectivity from $m_{\pi(k-1)}$ to m_{μ} , i.e. how much the mobilization of $m_{\pi(k-1)}$ favors the activation of m_{μ} . α, β are the parameters that control the relative weight of individual meme fitness and memetic synergy, respectively. Eq. 4.7 indicates that a meme is selected to form the memplex based not only on their individual performance, but also on how well they synergize with others. As a result, Eq. 4.7 prescribes that any subsequent candidate meme $m_{\pi(k)}$ for inclusion must have high fitness, as well as strong synergy with the mobilized meme $m_{\pi(k-1)}$.

When generating the memplex instance, meme selections ordering are made based on the $\{w_i, w_{ij}\}$ value judgements. We seek to include memes and arrange their mobilization orders that bring about the states of highest value, which obtain the greatest amount of reward for the memplex over the long run.

To balance between population-based search and individual learning in adaptive hybrids, a cognitive boundary $L_{\mathcal{M}}$ is enforced, to limit the size of elementary memes that compose the

memeplex candidate. This boundary is controlled through a simulated-annealing like rejection mechanism. For the candidate meme m_k obtained from Eq. 4.7, the candidate meme will be added to $\mathcal{M}_\pi$ with probability $P_{accept}(m_{\pi(k)})$ defined as:

$$P_{accept}(m_{\pi(k)}) = \begin{cases} 1 & w_{\pi(k)} \leq w_{\pi(k-1)} \\ \exp(-\frac{(w_{\pi(k-1)} - w_{\pi(k)})}{T}) & \text{otherwise} \end{cases} \quad (\text{Eq. 4.8})$$

where $T = \frac{1}{k-1}$. By Eq. 4.8, $m_{\pi(k)}$ will be added to $\mathcal{M}_\pi$ if it has a higher fitness than $m_{\pi(k-1)}$. Otherwise, a probability based on the fitness difference $(w_{\pi(k-1)} - w_{\pi(k)})$ and T is considered for the inclusion of $m_{\pi(k)}$ into $\mathcal{M}_\pi$. When the size of candidate memeplex $\varkappa = k$ is small, T will be sufficiently large, thus allowing diverse memes, i.e., even memes with lower fitness to possess some chance of participating in the memeplex. In this case, the lower meme fitness is compensated by its strong synergy exhibited in the previous step of Eq. 4.7. As the set of elementary memes increases, T decreases, imposing the constraint of mainly highly qualified candidate memes to be added into the memeplex. When a selected candidate meme m_k fails to be included in $\mathcal{M}$, the scheme terminates and the current $\mathcal{M}$ is returned as the memeplex instance. Eq. 4.8 enforces that if a meme is to be added to $\mathcal{M}_\pi$, its fitness must be justified for the additional computational cycles spent for their mobilization during the life cycle of $\mathcal{M}_\pi$.

From the algorithmic point of view, the individual learning discussed so far can be thought of as a self-organizing memetic process for emergence of adaptive memeplexes. This process consists of two simultaneous, interacting processes: the credit assignment process in the Sections 4.3.1 and 4.3.2 making the memeplex network topology $W_{\mathcal{M}}$ consistent with the individual memes and pattern of meme co-adaptation in memeplex (memeplex evaluation); and the memeplex assembling process making memeplex generation greedy with respect to the current network topology $W_{\mathcal{M}}$ (memeplex improvement). The two processes take place at both micro-level (i.e. at the level of individual meme) and macro-level (i.e., at the level of memeplex). In

the next section, we present a study on the proposed concept of memeplex for effective problem solving. In particular, the role of emergent memeplexes in the individual learning process of a memetic algorithm is investigated for solving the class of Capacitated Vehicle Routing Problem (CVRP).

4.4 Co-Adapted Memetic Algorithm for solving CVRP Problems

This section describes the Co-Adapted Memetic Algorithm (CAMA), an adaptive hybrid in which the memeplex is conceptually modeled as the adaptive individual learning for effective problem solving. The adaptive hybrid is proposed to solve the CVRP as detailed in Chapter 4. The basic steps of the proposed CMMA are outlined in Alg. 10. The algorithm starts with the initialization of a population $\text{Pop}(g = 0)$ of candidate solutions. At each generation, a population of offspring is generated using evolutionary search (i.e. crossover, mutation, split-repair) and subsequently undergoes individual learning defined by memeplex candidate $\mathcal{M}_\pi$ generated according to the *AssembleMemeplexCandidate* scheme. The individual learning process coupled with credit assignment schemes for individual memes and co-adapted memes, which performs the immediate and delayed rewarding as specified in Alg. 8, synchronizes the evolution of memeplexes according to the state of search. This workflow of the memetic search proceeds until some stopping conditions are reached, for example, the predefined maximum number of generations has elapsed.

Algorithm 10 Outline of CAMA Workflow

Input: Problem Instance**Output:** Global best solution s_g **Create** an initial population of solutions $\text{Pop}(g = 0) = \{s_1, s_2, \dots, s_i, \dots, s_M\}$ $W_{\mathcal{M}} = \{w_{ij}\}_{\kappa \times \kappa} \leftarrow$ **Create** a $\kappa \times \kappa$ matrix randomly**while** stopping conditions are not satisfied **do** $g \leftarrow g + 1$ **Perform** evolutionary operators to generate $\text{Pop}(gen)$ from $\text{Pop}(g - 1)$ based on $f(s_i)$ **Evaluate** $f(s_i), \forall s_i \in \text{Pop}(g)$ **for** $s_i \in \text{Pop}(g)$ **do** $t \leftarrow 0$

/* Adaptive memplex generation */

 $\mathcal{M}_{\pi} = \{m_{\pi(k)}\}_{k=1}^{k=\varkappa} \leftarrow \text{AssembleMemplexCandidate}(\Psi, W_{\mathcal{M}})$

/* Memplex solution refinement */

for $k = 1 \dots \varkappa$ **do**

/* Adaptive search intensity */

while $\text{IsMemplexIndividualLearningTerminated}(s_i, t, m_{\pi(k)})$ **do** $s_i \leftarrow \text{Refine}(s_i, m_{\pi(k)})$

/* Credit Assignment to individual memes */

 $w_{\pi(k)} \leftarrow \text{EvaluateMeme}(w_{\pi(k)}, \Delta f(s_i)_{m_{\pi(k)}}, \Delta T_{m_{\pi(k)}})$ $t \leftarrow t + 1$ **end while****end for**

/* Credit Assignment to co-adapted memes */

 $W_{\mathcal{M}} \leftarrow \text{EvaluateMemplex}(W_{\mathcal{M}}, \Delta f(s)_{\mathcal{M}_{\pi}}, \Delta T_{\mathcal{M}_{\pi}})$ **end for** $\text{Pop}(g) \leftarrow \text{Merge}(\text{Pop}(g), \text{Pop}(g - 1))$ **Update** global best solution s_g **end while**

The evolutionary operators in CAMA are designed to generate the genetic solutions solving the CVRPs [27], and include crossover, mutation, and split-repair operators. They have been described in details in Chapter 3. Here for completeness, we briefly described each of these evolutionary operators:

Crossover : In crossover, a child solution s_c is generated from partial routes of two selected parent solutions $s_i, s_j \in \text{Pop}(g)$ by slicing the VRP graph into two separate subgraphs containing non-empty customers, H_+ and H_- using a bisecting axis $\psi(\theta)$ which passes through the depot node e_0 as the axis center and combining the partial routes in $H_+^{s_i}$ and $H_-^{s_j}$. For details, refer to Section 3.4.1.1.

Mutation : The mutation operator is performed by one of the four mutation heuristics, namely *swap*, *inversion*, *insertion* and *displacement*, through random selection. Relatively high mutation rate is applied with a small population size for convergence reliability and velocity as well as population diversity for combinatorially explosive problems such as CVRP. For details, refer to Section 3.4.1.2.

Split-repair : The *Split-Repair* procedure is designed to revive any infeasible solution generated by either crossover or mutation operator, by removing the edge between consecutive customers in an infeasible route such that any increase in overall distance traveled due to the route split is minimized.

4.4.1 Memplex Representation for Vehicle Routing Problem

In the problems of vehicle routing, the memes that form memplex candidate $\mathcal{M}$ are taken from the meme terminal Ψ , and each meme m_k is defined by the meme tuple:

$$m_k = (\langle T_i \rangle, \langle \text{Acceptance_Strategy} \rangle, \langle \text{Search_Budget} \rangle)$$

With respect to the move operator T_i for solving VRPs, we define 6 move operators, which have been detailed in Section 3.4.2.1. In what follows, each of the move operators are briefly detailed:

T_1 : customer τ_{kj} of route $\tau_k \in s$ is removed and reinserted into a separate route $\tau_{k'} \in s, \tau_{k'} \neq \tau_k$, if fitness improvement $\Delta C(s) > 0$.

T_2 : customer τ_{kj} of route $\tau_k \in s$ is swapped with customer $\tau_{k'l}$ of route $\tau_{k'} \in s, \tau_{k'} \neq \tau_k$, if fitness improvement $\Delta C(s) > 0$.

T_3 : a sub-route $\hat{\tau}_k = \{\tau_{k(j+1)}, \tau_{k(j+2)}, \dots, \tau_{k(j+\mu)}\}$ of route $\tau_k \in s$ is swapped with sub-route $\hat{\tau}_{k'} = \{\tau_{k'(l+1)}, \tau_{k'(l+2)}, \dots, \tau_{k'(l+\nu)}\}$ of route $\tau_{k'} \in s, \hat{\tau}_k \neq \hat{\tau}_{k'}$, if fitness improvement $\Delta C(s) > 0$.

T_4 : two shorter routes $\tau_k, \tau_{k'}$ in the current solution are merged to reduce the number of routes and thus the overall distance traveled, if fitness improvement $\Delta C(s) > 0$.

T_5 : a customer $\tau_{kj} \in \tau_k$ is reinserted after a customer $\tau_{kl} \in \tau_k, \tau_{kj} \neq \tau_{kl}$, if fitness improvement $\Delta C(s) > 0$.

T_6 : a customer $\tau_{kj} \in \tau_k$ is joined with a customer $\tau_{kl} \in \tau_k, \tau_{kl} \neq \tau_{kj}$ by reversing the visiting order of customers between τ_{kj} and τ_{kl} , if fitness improvement $\Delta C(s) > 0$.

A meme terminal set Ψ is formed by assembling 24 memes from the combinatorial choices made among the 6 move operators, 2 acceptance strategies (*first improvement* or *best improvement*), and another 2 meme search intensity.

4.5 Empirical Study

In this section, an empirical study of CAMA for solving the CVRP is presented. A diverse set of CVRP benchmark classes as described in Subsection 3.3.1 are used. These CVRP classes differ

in size and other characteristics [42], including customer distribution, symmetric structure, route-length restriction, and vehicle capacity. The inclusion of a multitude of VRP benchmarks with wide-ranging characteristics renders the analysis of the proposed CAMA more thorough and rigorous. All the results presented in this section are summaries of 30 independent runs for each VRP instance under a consistent experimental setup. Each run continues until a maximum of 300,000 fitness evaluations is reached or the solution converged to the best known result in the literature. In each run, the evolutionary search configurations of the adaptive hybrids being studied for solving the benchmark problems are defined as follows: population size of 30, crossover probability of $p_c = 0.5$, mutation rate of $p_m = 0.5$, roulette-wheel selection scheme, and an elitism size of 2. Additionally, the adaptive memeplex individual learning in CAMA has the following parameter configurations: $\alpha = 1, \beta = 1, \rho = 0.999$. Student's t-test with 5% p -threshold significance level was conducted for assessing the statistical significance of the converged solution quality difference between two algorithms. Search results are then reported with respect to the following metrics:

- C - denotes the best fitness among the converged solutions on a VRP instance
- $\overline{C}$ - denotes the average fitness of the converged solutions on a VRP instance
- C_b - denotes the best known result of a VRP instance reported in the literature
- gap - denotes the minimal percentage fitness deviation from C_b as given by $\frac{C - C_b}{C_b} \times 100\%$
- $\overline{gap}$ - denotes the average percentage fitness deviation from C_b as given by $\frac{\overline{C} - C_b}{C_b} \times 100\%$
- t - denotes the computation time, in seconds, to arrive at the converged solution
- $\bar{t}$ - denotes the average computation time, in seconds, to arrive at the converged solutions
- g - denotes the average number of generations elapsed before the solution converged
- p_t - p -value denotes the *two-tailed* t-test difference between CAMA and the algorithm (5% p -threshold)
- p_r - p -value denotes the *right-tailed* t-test difference between CAMA and the algorithm (5% p -threshold)

4.5.1 Search Quality and Effectiveness of CAMA

To highlight the search efficiency and quality of CAMA, the results obtained are first pitted against two state-of-the-art adaptive hybrids, namely the Self-Generating Memetic Algorithm [96] (*SGMA*) and Multi-meme Memetic Algorithm [97] (*MultiMeme*) introduced in the literature, which are re-implemented here in the present work for possible assessment and comparison study.

The relative performances of the algorithms are depicted in Figs. 4.5, 4.6, 4.7, where the y-axis of sub-figures (a) and (b) denoting *gap* and $\overline{gap}$, respectively, and a common x-axis denoting the VRP instances of interest. It can be observed that on most problems, both MultiMeme and SGMA exhibit much larger gaps compared to CAMA, i.e., both the average and minimum gaps, indicating that CAMA is superior to MultiMeme MA and SGMA in terms of convergence to solutions that has fitness qualities nearer to both the best known fitness and average (expected) fitness of each respective VRP instances. To further validate the observed outcome, both the two-tailed and one-tailed t-tests are conducted and the detailed statistical results are presented in Tables 4.1, 4.2, 4.3, 4.4. The overall t-test results obtained confirmed that CAMA produces higher quality solutions than MultiMeme (exhibiting superiority on $\frac{2}{3}$ of the problem instances considered) and SGMA (superiority on half of the problem instances considered), outperforming these two algorithms *at 95% confidence level of statistical significance*.

$i(n)$	$algorithm$	$\bar{t}$ (seconds)	g	$\bar{C}$	$\overline{gap}$	C	gap	t-test on C		t-test on t	
								p_t	p_r	$p_b(T)$	$p_r(T)$
A32k5(32)	CAMA	0.13	73	784.00	0.00%	784.00	0.00%				
	SGMA	0.14	121	784.00	0.00%	784.00	0.00%	1.000	1.000	0.785	0.393
	MultiMeme	0.16	129	784.00	0.00%	784.00	0.00%	1.000	1.000	0.635	0.318
A54k7(54)	CAMA	1.62	729	1168.63	0.14%	1167.00	0.00%				
	SGMA	4.29	2496	1169.70	0.23%	1167.00	0.00%	0.195	0.097	0.046	0.023
	MultiMeme	2.23	632	1169.00	0.17%	1167.00	0.00%	0.726	0.363	0.531	0.265
A60k9(60)	CAMA	4.14	1655	1356.27	0.17%	1354.00	0.00%				
	SGMA	5.23	3156	1355.90	0.14%	1354.00	0.00%	0.699	0.651	0.405	0.202
	MultiMeme	2.52	1001	1359.00	0.37%	1354.00	0.00%	0.026 ⁽⁺⁾	0.013 ⁽⁺⁾	0.211	0.894
A69k9(69)	CAMA	5.38	1845	1162.97	0.34%	1159.00	0.00%				
	SGMA	7.63	3508	1163.80	0.41%	1159.00	0.00%	0.242	0.121	0.282	0.141
	MultiMeme	3.24	581	1164.58	0.48%	1159.00	0.00%	0.062	0.031	0.139	0.930
A80k10(80)	CAMA	9.33	2755	1771.50	0.48%	1763.00	0.00%				
	SGMA	10.06	3688	1776.57	0.77%	1763.00	0.00%	0.038 ⁽⁺⁾	0.019 ⁽⁺⁾	0.767	0.384
	MultiMeme	10.20	3849	1774.08	0.63%	1763.00	0.00%	0.472	0.236	0.796	0.398
B57k7(57)	CAMA	0.11	31	1140.00	0.00%	1140.00	0.00%				
	SGMA	0.16	60	1140.00	0.00%	1140.00	0.00%	1.000	1.000	0.057	0.029
	MultiMeme	0.27	85	1140.00	0.00%	1140.00	0.00%	1.000	1.000	0.141	0.070
B63k10(63)	CAMA	1.83	762	1536.90	2.73%	1526.00	2.01%				
	SGMA	2.72	1368	1537.33	2.76%	1537.00	2.74%	0.437	0.219	0.351	0.175
	MultiMeme	2.58	639	1539.25	2.89%	1537.00	2.74%	0.074	0.037	0.532	0.266
B78k10(78)	CAMA	9.25	2530	1225.37	0.36%	1221.00	0.00%				
	SGMA	12.36	5426	1230.23	0.76%	1221.00	0.00%	0.009 ⁽⁺⁾	0.004 ⁽⁺⁾	0.440	0.220
	MultiMeme	9.73	3126	1230.75	0.80%	1221.00	0.00%	0.065	0.033	0.893	0.446

Table 4.1: Comparison among CAMA, SGMA, MultiMeme on Augerat Benchmarks (⁽⁺⁾ indicates CAMA performs better statistically, in terms of p_t and p_r respectively)

$i(n)$	$algorithm$	$\bar{t}$ (seconds)	g	$\bar{C}$	$\overline{gap}$	C	gap	t-test on C		t-test on t	
								p_t	p_r	$p_b(T)$	$p_r(T)$
E76k7(76)	CAMA	8.20	2286	685.20	0.47%	682.00	0.00%				
	SGMA	6.19	2076	686.45	0.65%	685.00	0.44%	0.050 ⁽⁺⁾	0.025 ⁽⁺⁾	0.350	0.825
	MultiMeme	4.61	1047	686.55	0.67%	683.00	0.15%	0.108	0.054	0.111	0.945
E76k8(76)	CAMA	6.94	1906	739.33	0.59%	735.00	0.00%				
	SGMA	11.75	4564	738.47	0.47%	735.00	0.00%	0.311	0.845	0.077	0.039
	MultiMeme	6.81	1182	738.58	0.49%	736.00	0.14%	0.492	0.754	0.942	0.529
E76k10(76)	CAMA	8.79	2384	835.37	0.65%	830.00	0.00%				
	SGMA	11.41	4735	837.50	0.90%	832.00	0.24%	0.039 ⁽⁺⁾	0.019 ⁽⁺⁾	0.305	0.152
	MultiMeme	12.91	3838	840.67	1.29%	830.00	0.00%	0.040 ⁽⁺⁾	0.020 ⁽⁺⁾	0.189	0.094
E76k14(76)	CAMA	9.59	3502	1029.53	0.84%	1021.00	0.00%				
	SGMA	9.25	3988	1029.53	0.84%	1021.00	0.00%	1.000	0.500	0.872	0.564
	MultiMeme	7.03	1701	1031.92	1.07%	1021.00	0.00%	0.279	0.140	0.274	0.863

Table 4.2: Comparison among CAMA, SGMA, MultiMeme on CE Benchmarks, (⁽⁺⁾ indicates CAMA performs better statistically, in terms of p_t and p_r)

4.5.a: gap (the fitness gap in percentage between best fitness C and the best known fitness C_b)

4.5.b: $\overline{gap}$ (the fitness gap in percentage between average fitness $\overline{C}$ and the best known fitness C_b)

Figure 4.5: Minimum and Average percentage fitness gaps, gap , $\overline{gap}$, across Augerat & CE benchmarks (empty bars implies zero gap readings)

4.6.a: gap (the fitness gap in percentage between best fitness C and the best known fitness C_b)

4.6.b: $\overline{gap}$ (the fitness gap in percentage between average fitness $\overline{C}$ and the best known fitness C_b)

Figure 4.6: Minimum and Average percentage fitness gaps, $gap, \overline{gap}$, across Christofides benchmarks (empty bars implies zero gap readings)

4.7.a: gap (the fitness gap in percentage between best fitness C and the best known fitness C_b)

4.7.b: $\overline{gap}$ (the fitness gap in percentage between average fitness $\overline{C}$ and the best known fitness C_b)

Figure 4.7: Minimum and Average percentage fitness gaps, gap , $\overline{gap}$, across Golden benchmarks (empty bars implies zero gap readings)

$i(n)$	$algorithm$	$\bar{t}$	g	$\bar{C}$	$\overline{gap}$	C	gap	t-test on C		t-test on t	
		(seconds)						p_t	p_r	$p_b(T)$	$p_r(T)$
c1(50)	CAMA	1.81	745	526.01	0.27%	524.61	0.00%				
	SGMA	2.25	1162	525.38	0.15%	524.61	0.00%	0.382	0.809	0.609	0.304
	MultiMeme	0.72	367	525.82	0.23%	524.61	0.00%	0.843	0.578	0.117	0.941
c2(75)	CAMA	9.38	2989	840.36	0.61%	835.26	0.00%				
	SGMA	13.38	5426	842.15	0.82%	835.85	0.07%	0.077	0.039	0.101	0.050
	MultiMeme	7.52	1876	846.39	1.33%	841.48	0.74%	0.001(+)	0.000(+)	0.354	0.823
c3(100)	CAMA	23.14	4160	828.82	0.32%	826.14	0.00%				
	SGMA	16.02	3269	829.09	0.36%	826.14	0.00%	0.683	0.342	0.126	0.937
	MultiMeme	12.51	2160	831.09	0.60%	827.39	0.15%	0.033(+)	0.016(+)	0.057	0.972
c4(150)	CAMA	51.77	5019	1042.32	1.35%	1032.51	0.40%				
	SGMA	38.92	4760	1042.08	1.33%	1029.64	0.12%	0.880	0.560	0.075	0.962
	MultiMeme	44.27	2925	1051.65	2.26%	1033.74	0.52%	0.008(+)	0.004(+)	0.390	0.805
c5(199)	CAMA	90.12	5692	1318.92	2.13%	1299.52	0.62%				
	SGMA	75.82	6334	1322.56	2.41%	1307.86	1.27%	0.088	0.044	0.213	0.893
	MultiMeme	94.36	3401	1342.14	3.92%	1307.33	1.23%	0.015(+)	0.007(+)	0.696	0.348
c11(120)	CAMA	15.93	1806	1043.04	0.09%	1042.12	0.00%				
	SGMA	23.91	3588	1043.77	0.16%	1042.12	0.00%	0.139	0.069	0.085	0.042
	MultiMeme	15.36	1380	1044.43	0.22%	1042.97	0.08%	0.010(+)	0.005(+)	0.895	0.552
c12(100)	CAMA	0.84	122	819.56	0.00%	819.56	0.00%				
	SGMA	0.97	163	819.56	0.00%	819.56	0.00%	1.000	0.500	0.476	0.238
	MultiMeme	2.09	229	819.56	0.00%	819.56	0.00%	0.001(+)	0.000(+)	0.013	0.006

Table 4.3: Comparison among CAMA, SGMA, MultiMeme on Christofides Benchmarks, ⁽⁺⁾ indicates CAMA performs better statistically, in terms of p_t and p_r

$i(n)$	$algorithm$	$\bar{t}$ (seconds)	g	$\bar{C}$	$\bar{gap}$	C	gap	t-test on C		t-test on t	
								p_t	p_r	$p_b(T)$	$p_r(T)$
Pb1(240)	CAMA	111.47	2587	5794.87	2.97%	5667.34	0.71%				
	SGMA	142.60	4030	5796.54	3.00%	5659.87	0.57%	0.870	0.435	0.052	0.026
	MultiMeme	159.38	3863	5860.40	4.14%	5745.61	2.10%	0.018 ⁽⁺⁾	0.009 ⁽⁺⁾	0.056	0.028
Pb2(320)	CAMA	262.82	3462	8617.94	2.01%	8465.73	0.21%				
	SGMA	254.44	3627	8679.05	2.74%	8564.13	1.38%	0.003 ⁽⁺⁾	0.001 ⁽⁺⁾	0.762	0.619
	MultiMeme	301.64	3333	8864.83	4.94%	8569.58	1.44%	0.003 ⁽⁺⁾	0.001 ⁽⁺⁾	0.324	0.162
Pb3(400)	CAMA	332.78	2796	11301.85	2.41%	11092.21	0.51%				
	SGMA	344.02	3204	11348.33	2.83%	11051.63	0.14%	0.110	0.055	0.559	0.280
	MultiMeme	238.54	2593	11789.95	6.83%	11283.94	2.24%	0.004 ⁽⁺⁾	0.002 ⁽⁺⁾	0.064	0.968
Pb4(480)	CAMA	351.56	2260	13898.67	2.01%	13635.21	0.08%				
	SGMA	360.50	2240	13964.23	2.49%	13737.45	0.83%	0.027 ⁽⁺⁾	0.014 ⁽⁺⁾	0.570	0.285
	MultiMeme	348.74	1601	14428.63	5.90%	13942.37	2.33%	0.001 ⁽⁺⁾	0.000 ⁽⁺⁾	0.944	0.528
Pb5(200)	CAMA	30.29	1000	6492.39	0.49%	6460.98	0.00%				
	SGMA	41.93	1405	6480.75	0.31%	6460.98	0.00%	0.499	0.751	0.250	0.125
	MultiMeme	53.97	1654	6508.76	0.74%	6460.98	0.00%	0.566	0.283	0.112	0.056
Pb6(280)	CAMA	125.17	2146	8506.92	1.12%	8412.90	0.00%				
	SGMA	111.98	2129	8485.11	0.86%	8412.90	0.00%	0.188	0.906	0.543	0.729
	MultiMeme	191.25	3018	8502.27	1.06%	8412.90	0.00%	0.819	0.590	0.066	0.033
Pb7(360)	CAMA	244.66	2517	10342.62	1.44%	10195.59	0.00%				
	SGMA	268.79	3446	10393.00	1.94%	10232.47	0.36%	0.044 ⁽⁺⁾	0.022 ⁽⁺⁾	0.332	0.166
	MultiMeme	262.05	1862	10595.86	3.93%	10367.07	1.68%	0.001 ⁽⁺⁾	0.000 ⁽⁺⁾	0.691	0.345
Pb8(440)	CAMA	370.35	2553	12090.55	3.66%	11914.79	2.15%				
	SGMA	378.02	2667	12125.20	3.96%	11945.30	2.42%	0.188	0.094	0.599	0.300
	MultiMeme	377.25	1594	12593.86	7.98%	12118.56	3.90%	0.006 ⁽⁺⁾	0.003 ⁽⁺⁾	0.829	0.414
Pb9(255)	CAMA	154.86	5085	592.74	1.60%	585.71	0.40%				
	SGMA	164.56	6717	595.27	2.04%	588.49	0.87%	0.020 ⁽⁺⁾	0.010 ⁽⁺⁾	0.387	0.194
	MultiMeme	171.48	6668	595.98	2.16%	587.90	0.77%	0.072	0.036	0.488	0.244
Pb10(323)	CAMA	263.14	5807	756.26	1.92%	746.04	0.54%				
	SGMA	278.01	6478	760.88	2.54%	746.10	0.55%	0.008 ⁽⁺⁾	0.004 ⁽⁺⁾	0.362	0.181
	MultiMeme	213.23	6348	763.99	2.96%	751.13	1.23%	0.080	0.040	0.086	0.957
Pb11(399)	CAMA	367.13	5369	938.81	2.22%	925.08	0.72%				
	SGMA	373.91	5546	948.28	3.25%	932.28	1.51%	0.000 ⁽⁺⁾	0.000 ⁽⁺⁾	0.597	0.298
	MultiMeme	355.46	5700	947.63	3.18%	935.99	1.91%	0.006 ⁽⁺⁾	0.003 ⁽⁺⁾	0.647	0.677
Pb12(483)	CAMA	393.29	4204	1145.75	3.48%	1129.25	1.99%				
	SGMA	393.32	3707	1163.99	5.13%	1147.49	3.64%	0.000 ⁽⁺⁾	0.000 ⁽⁺⁾	0.997	0.499
	MultiMeme	356.90	3077	1232.13	11.28%	1139.84	2.95%	0.047 ⁽⁺⁾	0.023 ⁽⁺⁾	0.316	0.842
Pb13(252)	CAMA	136.38	6683	873.12	1.63%	864.93	0.68%				
	SGMA	134.09	7315	876.96	2.08%	870.04	1.27%	0.001 ⁽⁺⁾	0.001 ⁽⁺⁾	0.849	0.575
	MultiMeme	114.20	2194	907.58	5.64%	874.52	1.79%	0.001 ⁽⁺⁾	0.001 ⁽⁺⁾	0.322	0.839
Pb14(320)	CAMA	221.78	7358	1101.83	1.90%	1089.09	0.72%				
	SGMA	212.56	6671	1109.59	2.62%	1097.09	1.46%	0.000 ⁽⁺⁾	0.000 ⁽⁺⁾	0.625	0.688
	MultiMeme	170.64	2435	1172.85	8.47%	1100.57	1.78%	0.000 ⁽⁺⁾	0.000 ⁽⁺⁾	0.209	0.896
Pb15(396)	CAMA	341.27	7422	1370.54	1.88%	1361.20	1.19%				
	SGMA	305.59	6294	1377.28	2.38%	1365.19	1.48%	0.001 ⁽⁺⁾	0.000 ⁽⁺⁾	0.060	0.970
	MultiMeme	264.73	3901	1458.97	8.45%	1375.95	2.28%	0.004 ⁽⁺⁾	0.002 ⁽⁺⁾	0.171	0.914
Pb16(480)	CAMA	373.99	5784	1656.34	2.07%	1641.40	1.15%				
	SGMA	365.43	4830	1672.03	3.04%	1654.52	1.96%	0.000 ⁽⁺⁾	0.000 ⁽⁺⁾	0.593	0.704
	MultiMeme	248.67	1778	1895.89	16.84%	1664.49	2.58%	0.000 ⁽⁺⁾	0.000 ⁽⁺⁾	0.043	0.979
Pb17(240)	CAMA	145.49	6916	714.16	0.90%	710.41	0.37%				
	SGMA	107.90	6522	716.79	1.27%	709.21	0.20%	0.010 ⁽⁺⁾	0.005 ⁽⁺⁾	0.006	0.997
	MultiMeme	114.02	3777	726.95	2.71%	710.34	0.36%	0.007 ⁽⁺⁾	0.003 ⁽⁺⁾	0.129	0.936
Pb18(300)	CAMA	222.30	7549	1017.50	1.88%	1010.66	1.19%				
	SGMA	177.47	6524	1021.27	2.26%	1004.13	0.54%	0.004 ⁽⁺⁾	0.002 ⁽⁺⁾	0.010	0.995
	MultiMeme	191.47	5658	1048.43	4.98%	1016.78	1.81%	0.009 ⁽⁺⁾	0.004 ⁽⁺⁾	0.462	0.769
Pb19(360)	CAMA	290.17	7494	1388.94	1.62%	1377.15	0.75%				
	SGMA	311.02	7704	1396.43	2.16%	1382.74	1.16%	0.001 ⁽⁺⁾	0.001 ⁽⁺⁾	0.249	0.125
	MultiMeme	279.58	3059	1490.33	9.03%	1385.19	1.34%	0.001 ⁽⁺⁾	0.001 ⁽⁺⁾	0.840	0.580
Pb20(420)	CAMA	345.20	6924	1862.56	2.27%	1841.28	1.11%				
	SGMA	341.19	6170	1873.30	2.86%	1857.64	2.00%	0.000 ⁽⁺⁾	0.000 ⁽⁺⁾	0.836	0.582
	MultiMeme	358.29	4644	1936.88	6.35%	1852.93	1.75%	0.067	0.033	0.702	0.351

Table 4.4: Comparison among CAMA, SGMA, MultiMeme on Golden Benchmarks, ⁽⁺⁾ indicates CAMA performs better statistically, in terms of p_t and p_r

Next, we proceed to analyze the success rate of CAMA, MultiMeme and SGMA on each of the problem instance tabulated in Table 4.5. Here, success rate is measured as the percentage of the 30 independent runs that an algorithm converged to solution(s) at a precision level of 5% fitness gap of the best known result. Note that a $\checkmark$ symbol in Table 4.5 denotes a 100% success rate on a particular problem instance. It is observed that on the Augerat, CE and Christofides benchmarks, all three algorithms were able to achieve 100% success rate. On the Golden benchmark, which composes of large-scale VRP instances, however, the success rate is noted to drop significantly for MultiMeme and SGMA while CAMA is able to maintain a 100% success rate on all problems (with the exception of instance Pb18(300) only), highlighting that CAMA was able to perform much more robustly.

Benchmark	$i(n)$	Success Rate		
		CAMA	SGMA	MultiMeme
Augerat	A32k5(32)	✓	✓	✓
	A54k7(54)	✓	✓	✓
	A60k9(60)	✓	✓	✓
	A69k9(69)	✓	✓	✓
	A80k10(80)	✓	✓	✓
	B57k7(57)	✓	✓	✓
	B63k10(63)	✓	✓	✓
	B78k10(78)	✓	✓	✓
CE	E76k7(76)	✓	✓	✓
	E76k8(76)	✓	✓	✓
	E76k10(76)	✓	✓	✓
	E76k14(76)	✓	✓	✓
Christofides	c1(50)	✓	✓	✓
	c2(75)	✓	✓	✓
	c3(100)	✓	✓	✓
	c4(150)	✓	✓	✓
	c5(199)	✓	✓	66%
	c11(120)	✓	✓	✓
	c12(100)	✓	✓	✓
Golden	Pb1(240)	✓	✓	75%
	Pb2(320)	✓	✓	67%
	Pb3(400)	✓	✓	50%
	Pb4(480)	✓	✓	57%
	Pb5(200)	✓	✓	✓
	Pb6(280)	✓	✓	✓
	Pb7(360)	✓	✓	77%
	Pb8(440)	✓	97%	33%
	Pb9(255)	✓	✓	✓
	Pb10(323)	✓	✓	83%
	Pb11(399)	✓	97%	✓
	Pb12(483)	✓	77%	63%
	Pb13(252)	✓	✓	37%
	Pb14(320)	✓	✓	27%
	Pb15(396)	✓	✓	40%
	Pb16(480)	✓	✓	90%
	Pb17(240)	✓	✓	90%
	Pb18(300)	87%	53%	33%
	Pb19(360)	✓	83%	27%
	Pb20(420)	✓	✓	77%

Table 4.5: Success Rate of CAMA, SGMA, MultiMeme (✓ $\equiv$ 100%)

4.5.2 Computational Cost and Efficiency

In this section, we analyze the computational cost and efficiency of the proposed CAMA for solving CVRP with respect to adaptive MAs and the CMMA (as introduced in Chapter 3, which serves as baselines for comparison. Here the computational cost is measured based on the average wall clock time required for an algorithm in converging to solution(s) at a precision level of 5% fitness gap of the best known result for an instance.

To understand the impact of the proposed memeplex adaptation on the optimization process, we analyze the wall clock time for CAMA in converging to fitness within 5% *gap*, in comparison to adaptive MAs, including SGMA and MultiMeme. For most of the problem instances summarized in Figs. 4.8 and 4.9, CAMA takes approximately 0.1 seconds to converge, except on two large-scale problems (i.e., c4 and c5) where a few seconds is incurred. The results in Figs. 4.8, 4.9 also showed that CAMA consistently outperforms SGMA and MultiMeme in terms of the efficiency measure, indicating that SGMA and MultiMeme require more time budgets to converge to solutions of the same quality than CAMA on the representative VRP benchmark instances of diverse classes. This trend is more prominent on the large-scale problems as summarized in Fig. 4.10, where CAMA is noted to converge more efficiently than SGMA and MultiMeme by at least a factor of 2 and up to factor 10.

To gauge the computational cost imposed by the proposed memeplex adaptation in stochastic search, we assess the additional computational cost of CAMA by comparing it against the CMMA introduced in Chapter 3, which is a simple hybrid with *multi-memes* individual learning designed to exploit prior knowledge on the suitability of local search heuristics in CVRP [27]. Figs. 4.8 and 4.9, which summarized the results for small to medium-scale CVRPs, the computational cost of CAMA closely matches that of the CMMA, with only slight differences in the scale of 0.01 seconds. On the large-scale problems depicted in Fig. 4.10, the computational cost of CAMA also closely matches that of CMMA, trailing behind by one to ten

seconds. Notably CAMA appears to take less time to converge, as observed on several large-scale problems (e.g., Pb18 with 300 customers, Pb19 with 360 customers, and Pb20 with 420 customers).

Figure 4.8: Plot of average time required for convergence across Augerat and CE benchmarks

Figure 4.9: Plot of average time required for convergence across Christofides benchmarks

Figure 4.10: Plot of average time required for convergence across Golden benchmarks

4.5.3 CAMA against other recent state-of-the-art meta-heuristics algorithms in the literature

In this section, further comparative studies of the CAMA to recent advanced meta-heuristics is provided. In particular, we pit CAMA against 6 categories of meta-heuristics search algorithms previously developed for solving VRPs, which ranges from i) Adaptive Memetic Algorithms ii) VRP domain-specific Memetic Algorithm iii) Simulated Annealing iv) Tabu Search v) Genetic Algorithms to vi) VRP domain-specific Local Search as candidate stochastic search, on their respective set of benchmarks considered. In particular, CAMA is pitted against *GVR1*, *GVR2*, *GVR3* and *GVR4* on the Augerat and CE benchmarks, *BB*, *BR*, *CGL*, *GHL*, *GTS*, *OS*, *RR*, and *WH* on the Christofides benchmarks, and *BR*, *GTS*, *RTR*, *SEEA*, and *XK* on the Golden benchmarks. Overall, a total of 15 stochastic search algorithms are used as baselines for comparison in the present study, including two of the algorithms in the 1% club for the large-scale Golden benchmark set. (i.e., *BR* [209] and *SEEA* [170])

Tables 4.6, 4.7, and 4.8 summarized the results reported and published by the respective algorithms considered. The plots of gap and $\overline{gap}$ are depicted in Figs. 4.11, while on Figs. 4.12 and 4.13, the minimum fitness gap is reported. It is observed that CAMA achieved per-

<i>SEEA</i>	-	Memetic Algorithm proposed in [170] that uses specialized splitting procedure in place of trip delimiters.
<i>BB</i>	-	<i>Co-evolved Hybrid Genetic Algorithm</i> proposed in [11]
<i>OS</i>	-	Metastrategy Simulated Annealing algorithm proposed in [156]
<i>GHL</i>	-	<i>TabuRoute</i> proposed in [58]
<i>CGL</i>	-	<i>Tabu Search</i> proposed in [35]
<i>GTS</i>	-	<i>Granular Tabu Search</i> proposed in [216]
<i>RR</i>	-	<i>Parallel Tabu Search Algorithm</i> using <i>Ejection Chain</i> proposed in [174]
<i>BR</i>	-	<i>Boneroute</i> Memetic Algorithm proposed in [209]
<i>XK</i>	-	<i>Xu and Kelly Tabu Search</i> proposed in [231]
<i>GVR_x</i>	-	Genetic Algorithm with <i>Genetic Vehicle Representation</i> proposed in [162]
	($x = 1, 2, 3, 4$)	
<i>RTR</i>	-	<i>Record-to-Record Algorithm</i> proposed in [63]
<i>WH</i>	-	<i>Repeated Matching Heuristic</i> proposed in [225]

formances that are superior to most of the existing stochastic search algorithms considered (including the SEEA and BR which are in the 1% club for the large scale Golden benchmark set), in terms of both the best and average results, indicating that CAMA is able to produce both high quality and robust performance. It is notable that the reason behind the differing VRP benchmarks and performance metrics used when comparing CAMA with the respective advanced meta-heuristics considered, is a result of the different problem set(s) considered by the respective published works. Further, for the sake of fair comparison, we used the same performance metrics reported by earlier works when assessing the CAMA against the respective advanced meta-heuristics considered. The empirical results strongly indicates the efficiency of the memeplex adaptation mechanism of the proposed adaptive hybrid.

$i(n)$	algorithm	$\bar{C}$	$\overline{gap}$	C	gap	C_b
A32k5(32)	CAMA	784.00	0.00%	784.00	0.00%	784
	$GV R_1$	793.40	1.20%	784.00	0.00%	
	$GV R_2$	790.90	0.88%	784.00	0.00%	
	$GV R_3$	796.50	1.59%	784.00	0.00%	
	$GV R_4$	786.30	0.29%	784.00	0.00%	
A54k7(54)	CAMA	1168.63	0.14%	1167.00	0.00%	1167
	$GV R_1$	1178.60	0.99%	1167.00	0.00%	
	$GV R_2$	1186.20	1.65%	1167.00	0.00%	
	$GV R_3$	1182.40	1.32%	1167.00	0.00%	
	$GV R_4$	1188.90	1.88%	1167.00	0.00%	
A60k9(60)	CAMA	1356.27	0.17%	1354.00	0.00%	1354
	$GV R_1$	1377.90	1.77%	1358.00	0.30%	
	$GV R_2$	1377.90	1.77%	1354.00	0.00%	
	$GV R_3$	1387.90	2.50%	1358.00	0.30%	
	$GV R_4$	1372.40	1.36%	1358.00	0.30%	
A69k9(69)	CAMA	1162.97	0.34%	1159.00	0.00%	1159
	$GV R_1$	1180.60	1.86%	1159.00	0.00%	
	$GV R_2$	1182.00	1.98%	1165.00	0.52%	
	$GV R_3$	1180.00	1.81%	1165.00	0.52%	
	$GV R_4$	1179.90	1.80%	1165.00	0.52%	
A80k10(80)	CAMA	1771.50	0.48%	1763.00	0.00%	1763
	$GV R_1$	1811.20	2.73%	1764.00	0.06%	
	$GV R_2$	1813.80	2.88%	1777.00	0.79%	
	$GV R_3$	1819.30	3.19%	1781.00	1.02%	
	$GV R_4$	1811.00	2.72%	1774.00	0.62%	
B57k7(57)	CAMA	1140.00	0.00%	1140.00	0.00%	1140
	$GV R_1$	1141.10	0.10%	1140.00	0.00%	
	$GV R_2$	1141.20	0.11%	1140.00	0.00%	
	$GV R_3$	1141.70	0.15%	1140.00	0.00%	
	$GV R_4$	1140.70	0.06%	1140.00	0.00%	
B63k10(63)	CAMA	1536.90	2.73%	1526.00	2.01%	1496
	$GV R_1$	1546.60	3.38%	1507.00	0.74%	
	$GV R_2$	1544.00	3.21%	1496.00	0.00%	
	$GV R_3$	1547.00	3.41%	1516.00	1.34%	
	$GV R_4$	1545.70	3.32%	1497.00	0.07%	
B78k10(78)	CAMA	1225.37	0.36%	1221.00	0.00%	1221
	$GV R_1$	1266.00	3.69%	1233.50	1.02%	
	$GV R_2$	1254.60	2.75%	1224.00	0.25%	
	$GV R_3$	1249.50	2.33%	1223.00	0.16%	
	$GV R_4$	1252.30	2.56%	1224.00	0.25%	
E76k7(76)	CAMA	685.20	0.47%	682.00	0.00%	682
	$GV R_1$	705.90	3.50%	683.00	0.15%	
	$GV R_2$	704.80	3.34%	687.00	0.73%	
	$GV R_3$	703.40	3.14%	691.00	1.32%	
	$GV R_4$	705.00	3.37%	683.00	0.15%	
E76k8(76)	CAMA	739.33	0.59%	735.00	0.00%	735
	$GV R_1$	755.90	2.84%	737.00	0.27%	
	$GV R_2$	755.30	2.76%	738.00	0.41%	
	$GV R_3$	756.20	2.88%	738.00	0.41%	
	$GV R_4$	755.40	2.78%	740.00	0.68%	
E76k10(76)	CAMA	835.37	0.65%	830.00	0.00%	830
	$GV R_1$	851.70	2.61%	837.00	0.84%	
	$GV R_2$	856.60	3.20%	841.00	1.33%	
	$GV R_3$	852.60	2.72%	830.00	0.00%	
	$GV R_4$	854.00	2.89%	837.00	0.84%	
E76k14(76)	CAMA	1029.53	0.84%	1021.00	0.00%	1021
	$GV R_1$	1042.90	2.14%	1028.00	0.69%	
	$GV R_2$	1043.50	2.20%	1022.00	0.10%	
	$GV R_3$	1045.20	2.37%	1031.00	0.98%	
	$GV R_4$	1043.00	2.15%	1030.00	0.88%	

Table 4.6: Converged Solution Quality of the Algorithms on the Augerat and CE benchmarks, the best performing algorithm of the respective CVRP instance is highlighted in Bold

$i(n)$	algorithm	C	gap	C_b
c1(50)	CAMA	524.61	0.00%	524.61
	BB	524.61	0.00%	
	BR	524.61	0.00%	
	CGL	524.61	0.00%	
	GHL	524.61	0.00%	
	GTS	524.61	0.00%	
	OS	524.61	0.00%	
	RR	524.61	0.00%	
	WH	524.61	0.00%	
c2(75)	CAMA	835.26	0.00%	835.26
	BB	835.26	0.00%	
	BR	835.26	0.00%	
	CGL	835.45	0.02%	
	GHL	835.77	0.06%	
	GTS	838.60	0.40%	
	OS	844.00	1.05%	
	RR	835.32	0.01%	
	WH	835.80	0.06%	
c3(100)	CAMA	826.14	0.00%	826.14
	BB	827.39	0.15%	
	BR	826.14	0.00%	
	CGL	829.44	0.40%	
	GHL	829.45	0.40%	
	GTS	828.56	0.29%	
	OS	838.00	1.44%	
	RR	827.53	0.17%	
	WH	830.70	0.55%	
c4(150)	CAMA	1032.51	0.40%	1028.42
	BB	1036.16	0.75%	
	BR	1030.88	0.24%	
	CGL	1038.44	0.97%	
	GHL	1036.16	0.75%	
	GTS	1033.21	0.47%	
	OS	1044.35	1.55%	
	RR	1044.35	1.55%	
	WH	1038.50	0.98%	
c5(199)	CAMA	1299.52	0.62%	1291.45
	BB	1324.06	2.53%	
	BR	1314.11	1.75%	
	CGL	1305.87	1.12%	
	GHL	1322.65	2.42%	
	GTS	1318.25	2.08%	
	OS	1334.55	3.34%	
	RR	1334.55	3.34%	
	WH	1321.30	2.31%	

$i(n)$	k	C	gap	C_b
c11(120)	CAMA	1042.12	0.00%	1042.11
	BB	1043.11	0.10%	
	BR	1042.11	0.00%	
	CGL	1074.13	3.07%	
	GHL	1073.47	3.01%	
	GTS	1042.87	0.07%	
	OS	1043.00	0.09%	
	RR	1042.11	0.00%	
	WH	1043.40	0.12%	
c12(100)	CAMA	819.56	0.00%	819.56
	BB	819.56	0.00%	
	BR	819.56	0.00%	
	CGL	819.56	0.00%	
	GHL	819.56	0.00%	
	GTS	819.56	0.00%	
	OS	819.59	0.00%	
	RR	819.56	0.00%	
	WH	819.60	0.00%	

(a) c1-c5

(b) c11-c12

Table 4.7: Converged Solution Quality of the Algorithms on Christofides benchmarks, the best performing algorithm of the respective CVRP instance is highlighted in Bold

$i(n)$	algorithm	C	gap	C_b	$i(n)$	k	C	gap	C_b
Pb1(240)	CAMA	5667.34	0.71%	5627.54	Pb11(399)	CAMA	925.08	0.72%	918.45
	BR	5676.97	0.88%			GTS	936.04	1.92%	
	GTS	5736.15	1.93%			RTR	934.33	1.73%	
	RTR	5834.60	3.68%			SEEA	939.74	2.32%	
	SEEA	5648.04	0.36%			XK	932.68	1.55%	
Pb2(320)	CAMA	8465.73	0.21%	8447.92	Pb12(483)	CAMA	1129.25	1.99%	1107.19
	BR	8512.64	0.77%			GTS	1147.14	3.61%	
	GTS	8553.03	1.24%			RTR	1137.18	2.71%	
	RTR	9002.26	6.56%			SEEA	1152.88	4.13%	
	SEEA	8459.73	0.14%			XK	1140.72	3.03%	
Pb3(400)	CAMA	11092.21	0.51%	11036.23	Pb13(252)	CAMA	864.93	0.68%	859.11
	BR	11199.72	1.48%			GTS	868.80	1.13%	
	GTS	11402.75	3.32%			RTR	881.04	2.55%	
	RTR	11879.95	7.65%			SEEA	877.71	2.17%	
	SEEA	11036.23	0.00%			XK	881.07	2.56%	
Pb4(480)	CAMA	13635.21	0.08%	13624.53	Pb14(320)	CAMA	1089.09	0.72%	1081.31
	BR	13637.53	0.10%			GTS	1096.18	1.38%	
	GTS	14910.62	9.44%			RTR	1103.69	2.07%	
	RTR	14639.32	7.45%			SEEA	1089.93	0.80%	
	SEEA	13728.80	0.77%			XK	1118.09	3.40%	
Pb5(200)	CAMA	6460.98	0.00%	6460.98	Pb15(396)	CAMA	1361.20	1.19%	1345.23
	BR	6460.98	0.00%			GTS	1369.44	1.80%	
	GTS	6697.53	3.66%			RTR	1364.23	1.41%	
	RTR	6702.73	3.74%			SEEA	1371.61	1.96%	
	SEEA	6460.98	0.00%			XK	1377.79	2.42%	
Pb6(280)	CAMA	8412.90	0.00%	8412.88	Pb16(480)	CAMA	1641.40	1.15%	1622.69
	BR	8429.28	0.19%			GTS	1652.32	1.83%	
	GTS	8963.32	6.54%			RTR	1657.93	2.17%	
	RTR	9016.93	7.18%			SEEA	1650.94	1.74%	
	SEEA	8412.90	0.00%			XK	1656.66	2.09%	
Pb7(360)	CAMA	10195.59	0.00%	10195.56	Pb17(240)	CAMA	710.41	0.37%	707.79
	BR	10216.50	0.21%			GTS	711.07	0.46%	
	GTS	10547.44	3.45%			RTR	720.44	1.79%	
	RTR	11213.31	9.98%			SEEA	717.09	1.31%	
	SEEA	10267.50	0.71%			XK	747.24	5.57%	
Pb8(440)	CAMA	11914.79	2.15%	11663.55	Pb18(300)	CAMA	1010.66	1.19%	998.73
	BR	11936.16	2.34%			GTS	1016.83	1.81%	
	GTS	12036.24	3.20%			RTR	1029.21	3.05%	
	RTR	12514.20	7.29%			SEEA	1018.74	2.00%	
	SEEA	11865.40	1.73%			XK	1066.57	6.79%	
Pb9(255)	CAMA	585.71	0.40%	583.39	Pb19(360)	CAMA	1377.15	0.75%	1366.86
	GTS	593.35	1.71%			GTS	1400.96	2.49%	
	RTR	587.09	0.63%			RTR	1403.05	2.65%	
	SEEA	596.89	2.31%			SEEA	1385.60	1.37%	
	XK	589.10	0.98%			XK	1435.88	5.05%	
Pb10(323)	CAMA	746.04	0.54%	742.03	Pb20(420)	CAMA	1841.28	1.11%	1821.15
	GTS	751.66	1.30%			GTS	1915.83	5.20%	
	RTR	749.15	0.96%			RTR	1875.17	2.97%	
	SEEA	751.41	1.26%			SEEA	1846.55	1.39%	
	XK	746.56	0.61%			XK	1934.99	6.25%	

(a) Pb1-Pb10

(b) Pb11-Pb20

Table 4.8: Converged Solution Quality of the Algorithms on Golden benchmarks, the best performing algorithm of the respective CVRP instance is highlighted in Bold

 4.11.a: Minimum fitness *gap*

 4.11.b: Average fitness *gap*

 Figure 4.11: Minimum and Average percentage fitness gaps, gap , $\overline{gap}$, across Augerat and CE benchmarks

4.12.a: Problem Instances 1-8

4.12.b: Problem Instances 9-14

100
4.12.c: Problem Instances 15-20

Figure 4.12: Minimum percentage fitness gaps, gap , across Golden benchmarks

Figure 4.13: Minimum percentage fitness gaps, *gap*, across Christofides benchmarks (c1 and c12 omitted due to their insignificant performance)

4.6 Conclusions

In this chapter, an adaptive hybrid with the conceptual modeling of memeplexes in search as a set of co-adapted memes has been introduced. In particular, the proposed memeplex representation, credit assignment criteria for evaluating individual memes and meme co-adaptation, and the role of emergent memeplexes in the individual learning process of a memetic algorithm have been described in great details. Subsequently, the Co-Adapted Memetic Algorithm (CAMA) as a manifestation of the proposed conceptual modeling of memeplexes for solving CVRPs of diverse characteristics is then proposed and studied empirically on four sets of VRP benchmarks with diverse characteristics. Results obtained on the VRP benchmarks demonstrated that CAMA not only produced higher quality solutions compared to MultiMeme on $\frac{2}{3}$ of the problem instances, but also superior to SGMA on half of the problem instances considered at 95% confidence level of statistical significance. In terms of scalability, on the Golden benchmark which contains many large-scale VRP instances, CAMA also displayed a near 100% success rate, in contrast to MultiMeme and SGMA where significant degradation in the

success rate is observed. Furthermore, in terms of wall clock time for solution convergence, CAMA is noted to be more efficient than SGMA and MultiMeme and closely matches the performances of CMMA (introduced in Chapter 3, which was crafted using prior knowledge available on the suitability of memes for the design of multi-memes individual learning). Last but not least, CAMA achieved performances that are superior to most of the existing stochastic search algorithms considered for solving CVRP (including the SEEA and BR which are in the 1% club for the large scale Golden benchmark set). In summary, the conceptual modeling of meme complexes in search, through the CAMA for solving on a plethora of VRPs, is shown to be capable of attaining high quality and efficient performance efficaciously, when being assessed against several recent adaptive hybrids and human-designed state-of-the-art advanced metaheuristics.

Chapter 5

A Memeplex Robust Search Scheme

In this chapter, the memeplex concept is extended and demonstrated for solving problem environments with uncertainty. In particular, the *Stochastic-Adaptive Memeplex Robust Search* (SAMRS) is proposed for finding robust and reliable solutions that are less sensitive to stochastic behaviors of customer demands and have low probability of route failures, respectively, in the context of *Vehicle Routing Problem with Stochastic Demands* (VRPSD). The approach involves a memeplex representation and a gene-meme coevolution model with Robust Solution Search Scheme in memetic computation for the optimization problems of VRPSD.

5.1 Introduction

Although classical Operational Research problems such as CVRPs are typically treated without stochastic elements [144], the real-world scenarios in practice have to accept that most of the information and data about the operating environment such as customer demands cannot be known beforehand, thus the increasing interests on problem models that are dynamic, stochastic, etc. This suggests the need for innovative ways to attain effective and robust solutions that address problem environment with uncertainties like the VRPSD, efficiently. The proposed *Stochastic-Adaptive Memeplex Robust Search* (SAMRS) finds robust and reliable solutions that are less sensitive to stochastic behaviors of problem model. In particular, the contribution of this work is three-fold. First, the proposed SAMRS employs the *Robust Solution Search*

Scheme (RS^3) as an approximation of the computationally intensive Monte Carlo simulation, thus reducing the computation cost of simulation-based fitness evaluation in problems with uncertainty, while directing the search towards robust and reliable solutions. Furthermore, an adaptive individual learning based on the conceptual modeling of memeplex is introduced in the SAMRS. Finally, SAMRS incorporates a gene-meme coevolution model with genetic and memetic representation to effectively manage the search for solutions in VRPSD. Extensive experimental results are then presented for benchmark problems to demonstrate that the proposed SAMRS serves as an efficacious means of generating high-quality robust and reliable solutions in VRPSD.

The rest of the chapter is organized as follows: Section 5.2 begins with an introduction of the VRPSD. Subsequently, Section 5.3 discusses SAMRS with respect to the genetic and memetic representations, fitness evaluation, individual learning using memeplex, and the gene-meme co-evolution model, for finding robust and reliable solutions. In Section 5.4, the empirical assessment of *SAMRS* on extensive numerical experiments is considered. The superior results obtained by *SAMRS* in the experiments indicated that it serves as an efficacious means of generating high-quality robust and reliable solutions in VRPSD. Finally, the last section ends with a brief conclusion of the present research.

5.2 Vehicle Routing Problem with Stochastic Demands

The Vehicle Routing Problem with Stochastic Demands (VRPSD) involves a set of customers with stochastic demands that must be served by a fleet of vehicles with routes starting and ending at a depot. As introduced previously in Chapter 3, the VRP is considered as one of the most difficult NP-hard problems due to its complex combinatorial nature, which is the fusion of two NP-hard problems, namely the Traveling Salesman Problem (TSP) and the Bin Packing Problem (BPP). In contrast to the classical VRP, VRPSD is deemed as an even more

complex problem due to the uncertainty introduced into VRP by the stochastic demands of the customers.

To solve VRPSD, [188] classified the existing approaches into *dynamic* or *static* class. The Markov Decision Process (MDP) approach of [188, 34, 114], the neuro-dynamic programming approach of [187], etc., are examples of dynamic approaches. In these approaches, routing decisions are made in multiple stages and based on demand realizations. Routes are re-optimized at each stage based on the remaining vehicle capacity and the set of unvisited customers. For routing decisions in static approaches such as chance constraint programming [190], robust optimization [199], and stochastic programming with recourse [30], once the route decisions are made, they remain unchanged irregardless of the demand realizations. Generally, static strategies have been preferred over their dynamic counterparts because static strategies are useful when a stable reliable solution is desired [13] or when re-optimization during route execution is impossible, due to the lack of information [119].

Among the static approaches, most VRPSD research assumes an *a priori* solution approach. In the first stage, complete *a priori* routes are designed before any actual demands become known. In the second stage, routes are traversed, demands are revealed, and extra trips to the depot for replenishment are performed if a customer's demand exceeds current vehicle capacity. The routes order are thus left unchanged. The objective is to find a route sequence that minimizes total expected cost from the original distance traveled and from extra trips to and from the depot. Exact approaches include that of [13] who proposed to construct an *a priori* sequence among all customers of minimal expected total length. [213] developed the *a priori* sequence based on simulated annealing. [104] then proposed a branch-and-bound method based on the integer L-shaped algorithm. Moreover, [15] used the length of the *a priori* tour to approximate the objective, and analyzed several well known metaheuristics for VRPSD. It verified metaheuristics as an effective approach to tackle vehicle routing problems with uncertain demands. [50] presented an evolutionary approach for routing problem with

stochastic demands, where a memetic algorithm for VRPSD is introduced. [143] extended a recourse strategy which allows vehicles to serve additional customers from failed routes prior to returning to the depot or to serve customers from failed routes using a new route after returning to the depot. In order to minimize costs further, [234] designed *a priori* routes that may prescribe returns to the depot before the vehicle capacity is depleted (i.e., a form of preventive restocking).

Due to the stochastic behavior of customer demands, the feasible solution designed in the first stage may turn out to be infeasible if the final demand of any route exceeds the actual vehicle capacity. This situation is referred to as “route failure” [85], and when it occurs, some corrective actions must be introduced to obtain a new feasible solution, which would cause increase in routing cost and control in reality. To address this situation, some authors proposed to construct reliable solutions for VRPSD, which aims at reducing the probability of occurrence on such undesirable situation. For instance, [202] described a multi-objective evolutionary algorithm to solve the capacity and time constrained VRPSD, in which the cost associated with the route failures has been considered. More recently, [85] introduced the idea to keep a certain amount of surplus vehicle capacity (safety stock) while designing the routes so that if the final routes’ demands exceed their expected values up to a certain limit, they can be satisfied without incurring a route failure.

5.2.1 Problem Formulation for VRPSD

In the formulation of VRPSD, each customer e_l has a stochastic $demand(e_l)$ that must be serviced by vehicle v_k , customer demand $demand(i)$ is only revealed at each stop of customer i . During the actual distribution process, the route τ_k may fail to meet the capacity constraint at its j^{th} customer when $C_v < \sum_{j=1}^m demand(\tau_{kj})$, at which point v_k will have to take a recourse action from $\tau_{k(j-1)}$ to the depot and then back to τ_{kj} .

In this context, the objective of the VRPSD is to find robust solution $s = \{\tau_1, \tau_2, \dots, \tau_k, \dots, \tau_K\}$, that satisfies all customer demands and vehicle capacity constraint C_v , and at the same time minimizes the overall expected distance traveled by all vehicles, $C_{VRPSD}(s)$ as given by:

$$C_{VRPSD}(s) = \sum_{k=1}^K L_{VRPSD}(\tau_k) \quad (\text{Eq. 5.1})$$

where $L_{VRPSD}(\tau_k)$ is the expected distance traveled by vehicle v_k . The solution s is robust [112] in the sense that its value does not deviate abruptly as a result of the stochastic behaviors of customer demands in VRPSD. If a solution obtained for VRPSD is highly sensitive to small perturbation in customer demands, its performance becomes unreliable for use in VRPSD [217].

5.2.2 Safety Stock for Managerial Control over Route Failure in VRPSD

The expected cost of routing in a stochastic setting may not be the sole objective in VRPSD. Other goals may include minimizing route failures during the actual distribution process. This is because whenever a route failure occurs, the vehicle with larger-than-expected accumulated demand has to return to the depot for a refill, which tends to increase the managerial cost.

The concept of safety stock [85] can be applied to minimize the number of route failures during the actual distribution process. The basic principle behind the safety stock level is that only a certain fraction k of the vehicle total capacity C_v will be considered as available during the routing design phase. In other words, during the VRPSD objective evaluation, the vehicle is modeled to have a lower vehicle capacity C_i , as given by:

$$C_i = k \cdot C_v \quad (\text{Eq. 5.2})$$

where C_v denotes vehicle capacity of the original VRPSD instance. The parameter $k \in (0, 1)$ is known as the safety stock level [85]. In the routing design phase, the capacity constraint on solution s is then defined as

$$C_v \geq \sum_{i \in \tau_k} demand(i), \forall \tau_k \in route(s) \quad (\text{Eq. 5.3})$$

Here, the objective is to construct reliable solutions (i.e., solutions with a low probability of suffering route failures) with optimal or pseudo-optimal total expected costs under the random behaviors of customer demands. This is based on the concept of safety stock which has been previously applied for VRPSD to ensure the managerial control over the frequency of route failure and expected cost penalty [105, 85]. Control over route failures provided by the safety stock level k is effective in that routes will remain feasible as long as the extra demand that might be originated in each route during the routing execution time does not exceed the vehicle reserve capacity. This vehicle reserve capacity, or *safety stock*, is given by:

$$\text{safety stock} = (1 - k) \cdot C_v \quad (\text{Eq. 5.4})$$

where k denotes the safety stock level associated with routing solution s . The solution reliability can be measured by the probability that a solution route will suffer from failure (i.e., capacity constraint violation) during the actual distribution phase.

5.3 Memplex Robust Search Methodology

In this section, the *Stochastic-Adaptive Memplex Robust Search* (SAMRS) is proposed as a memplex framework for solving problems with uncertainty such as VRPSD. SAMRS is a general approach for optimization in uncertain environments which aims at tackling the problems with globally optimal solutions of high tolerance or robustness to uncertainties, at the same time without sacrificing the search capability.

The genetic-memetic representation, RS^3 fitness evaluation, adaptive memplex individual learning, a gene-meme coevolution model that forms the basis of SAMRS. The genetic solution models a problem solution encoded in the genetic representation. For the VRPSD studied,

this genetic representation involves the vehicle routes and the safety stock level. On the other hand, the memeplex denotes a memetic recipe which encodes the mutually assisting memes that self-adapts to effectively refine its affiliated genetic solution during the individual learning of SAMRS. Furthermore, The robust solution search scheme (RS^3) provides a fitness prediction model that effectively induces the spirit of Baldwinian learning [152] within SAMRS, by inducing perturbations into the phenotypic features of the problem solutions, thus directs the search towards robust solutions. The population-based search of SAMRS is guided by a gene-meme coevolution model that co-adapts the genetic evolution of geneplexes (i.e., problem solutions) along with the memetic evolution of memeplexes (i.e. refinement solvers). Fig. 5.1 depicts the various components of SAMRS, including solution representation, fitness evaluation, adaptive memeplex individual learning scheme, gene-meme coevolution (e.g., genetic/memetic crossover and mutation), and genetic representation repair. The genetic representation repair procedure is used to restore any infeasible solution representation in the geneplex solutions before fitness evaluation. In the subsequent sections, the proposed SAMRS with respect to the designs of its various search components as depicted in Fig. 5.1 are described.

Figure 5.1: Outlines of SAMRS and its components

5.3.1 Genetic and Memetic Representations

Here the genetic representation s_i and memetic representation $\mathcal{M}_i$ in SAMRS is described. The genetic representation s_i denotes a solution point in parametric design space of a problem (e.g.,

the genetic representation, s_i in VRPSD includes routing design $\{\tau_1, \tau_2, \dots, \tau_k, \dots, \tau_K\}$ and safety stock level k_i associated). On the other hand, the memetic representation $\mathcal{M}_i$ denotes a memeplex, which comprises of a memetic strategy matrix W_i that guides the local refinement on the affiliated solution s_i during individual learning. The relationship of genetic solution and its associated memeplex is that of a cooperative co-evolution, in which memeplex attempts to improve the affiliated genetic solution, i.e., the routing solution. During the co-evolution, the genetic solution itself serves as the propagation vehicle for its memetic materials, and its survival and subsequent reproduction assures the survival and reproduction of both the high quality memetic and genetic materials through selection pressure.

In SAMRS, the encapsulation of genetic and memetic representations allows an individual $agt_i = \{s_i, \mathcal{M}_i\}$ in the population to behave more or less like an autonomous agent during the evolutionary and individual learning phases, whose behaviors manifest in the following aspects:

- *Autonomous*: In SAMRS, $s_i \in agt_i$ is capable of evaluating its fitness under the uncertainty in environment via Monte Carlo simulation; $s_i \in agt_i$ also reacts to stochastic customer demand in VRPSD during Monte Carlo simulation through the safety stock level and the statistical demand distribution function; $\mathcal{M}_i \in agt_i$ interprets and execute local search instructions (a.k.a., memes) from the meme pool to improve the routing solution during individual learning.
- *Learning*: In SAMRS, agt_i is capable of learning and improving the programming structure. Its associated memeplex $\mathcal{M}_i$ is a rule-based program, which self-configures through adaptive mechanism during individual learning. During the gene-meme coevolution, agt_i imitates and exchanges information (e.g., in VRPSD, on its routing solution, associated safety stock level and memeplex), with other individuals in the population $\text{Pop}(g)$ during the evolutionary phase.

- *Reactive*: In SAMRS, agt_i acts like a reactive subsystem. Its associated memeplex $\mathcal{M}_i$ interact with its own coupled genetic environment which consists of the other individuals $\text{Pop}(g)$ and the problem environment and provides it the evaluative feedback in terms of credit assignment and evolution selection pressure.
- *Proactive*: In SAMRS, agt_i has explicit goal/objective. Its associated genetic solution s_i can sense aspects of their genetic environment via $\text{Pop}(g)$ and Monte Carlo simulation routine; its associated memeplex $\mathcal{M}_i$ can choose local refinement actions via self-configuration(e.g. modifying $\mathcal{M}_i$'s memetic strategy matrix W_i) to influence their genetic environment (via individual learning in SAMRS). It is assumed in SAMRS from the beginning that agt_i has to operate (e.g. performs self-adaptive individual learning and gene-meme coevolution) despite significant uncertainty about the genetic environment.

Fig. 5.2 depicts the genetic and memetic representation defined in SAMRS, i.e., genotype and phenotype in genetics, while memotype and sociotype in memetics, respectively.

Figure 5.2: Genetic and Memetic Representations of $agt_i = \{s_i, \mathcal{M}_i\}$: s_i with the decoded routes, safety stock level k_i and $\mathcal{M}_i$ with search instructions

5.3.2 Fitness Evaluation based on Robust Solution Search Scheme

The Robust Solution Search Scheme (RS^3) is designed for convergence to robust solutions with the stochastic demands modeled as perturbations induced into the phenotypic representation of solutions before fitness evaluation [217]. Supported by both mathematical models and empirical results in [217], the RS^3 is used in SAMRS towards convergence to robust solutions for problems with uncertainty. Making little assumption on the nature of problem uncertainty such as the customer demand distribution in VRPSD, the scheme effects implicit averaging of solution fitness [217], i.e., averaging the fitness evaluations of each solution vector s and its perturbed vectors s^g across multiple generations g , which equates to a Monte Carlo simulation on vector s over time. In this case each return from RS^3 is an independent, identically distributed estimate of solution fitness. By the law of large numbers the sequence of averages of these estimates converges to their expected value, with each average itself as an unbiased estimate. In other words, RS^3 in SAMRS serves as an approximation of the Monte Carlo simulation that evolves towards a true Monte Carlo simulation with increasing generations. It thus constitutes as an efficient substitution of the computationally intensive simulation-based fitness evaluation, especially when dealing with large number of individual solutions produced during evolution search.

For the VRPSD being solved, in many real-world scenarios of the problem, historically archived data can be used to model the $PDF(demand(i), \mu_i, \sigma_i^2)$ of each stochastic customer demand. According to [85], Log-Normal distribution serves as a natural choice for modeling non-negative customer demands. Thus the Log-Normal distribution as the statistical model for customer demands is considered in the present study (although other distribution may also apply), which is defined as:

$$PDF_X(x, \mu_i, \sigma_i^2) = \frac{1}{x\sigma_i\sqrt{2\pi}} e^{-\frac{(\ln x - \mu_i)^2}{2\sigma_i^2}}, X = \{x | x = demand(i), 1 \leq i \leq N\} \quad (\text{Eq. 5.5})$$

$$\mu_i = \ln(E[demand(i)]) - \frac{1}{2} \ln\left(1 + \frac{\text{Var}[demand(i)]}{E[demand(i)]^2}\right), \quad (\text{Eq. 5.6})$$

$$\sigma_i^2 = \ln\left(1 + \frac{\text{Var}[demand(i)]}{E[demand(i)]^2}\right). \quad (\text{Eq. 5.7})$$

where $E[demand(i)]$ denotes the expected demand of the VRPSD instance and $\text{Var}[demand(i)]$ denotes the demand variance, which can be modeled as:

$$\text{Var}[demand(i)] = \alpha \cdot E[demand(i)] \quad (\text{Eq. 5.8})$$

with $\alpha \in (0, 1)$ denoting the variance level. The fitness of solution s_i is evaluated together with its respective perturbed problem instances $\{P_i\}_{i=1}^I$ which is given by:

$$f(s_i) = \frac{\sum_{i=1}^I f_i(s_i)}{I} \quad (\text{Eq. 5.9})$$

$$= \frac{\sum_{i=1}^I \sum_{k=1}^K L_i(\tau_k)}{I} \quad (\text{Eq. 5.10})$$

$$= \frac{\sum_{i=1}^I \sum_{k=1}^K [\sum_{j=1}^{m-1} d_i(\tau_{kj}, \tau_{k(j+1)}) + d_i(\tau_{k1}, 0) + d_i(0, \tau_{km})]}{I} \quad (\text{Eq. 5.11})$$

under capacity constraint

$$\sum_{j=1}^m demand_i(\tau_{kj}) \leq C_i \equiv k_i \cdot C_v, \forall \tau_k \in \text{route}(s), i = 1, \dots, I \quad (\text{Eq. 5.12})$$

where $f_i(s_i)$ is the fitness of s_i as evaluated in the perturbed problem instance P_i , and $L_i(\tau_k)$ measures the travelled distance of route τ_k as evaluated in P_i . $d_i(\tau_{kj}, \tau_{k(j+1)})$ is the distance between two consecutive customers τ_{kj} and $\tau_{k(j+1)}$ in P_i , while $demand_i(\tau_{kj})$ is the demand of customer τ_{kj} as sampled in P_i . C_v denotes vehicle capacity of the VRPSD instance. Finally parameter $k_i \in (0, 1)$ is the safety stock level associated with solution s_i .

5.3.3 Individual Learning using Memeplex

In this section, the proposed memeplex individual learning in SAMRS is introduced. A memeplex is essentially composed of mutually assisting memes [40], that work together to achieve more than what each meme could accomplish alone [192, 16]. During the individual learning, the memeplex $\mathcal{M}_i$ manifests as a set of instructions or memes that carry out local refinement on its associated genetic solution s_i .

To significantly reduce the computational cost associated with a Monte-Carlo simulation fitness evaluation, individual learning process in SAMRS uses RS^3 for solution fitness evaluation after a solution is refined at each individual learning iteration. This dynamics induced by the problem uncertainty change the evaluation of a neighboring solution at each step of a local search. Consequently, instead of directly using the memeplex individual learning in CAMA from Chapter 4 (which is more appropriate for solving exact problem in which uncertainty does not exist), individual learning in SAMRS implements handle such dynamics in fitness evaluation. In other words, an individual memeplex $\mathcal{M}_i$ in SAMRS learns to adjust its search neighborhood to its affiliated solution s_i at each step in an individual learning.

The structure of each memeplex $\mathcal{M}_i$ in SAMRS is represented by its very own strategy matrix W_i , which is given by:

$$W_i = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1\kappa} \\ w_{21} & w_{22} & \cdots & w_{2\kappa} \\ \vdots & \vdots & \ddots & \vdots \\ w_{\kappa 1} & w_{\kappa 2} & \cdots & w_{\kappa \kappa} \end{pmatrix} \quad (\text{Eq. 5.13})$$

where κ denotes the size of the meme terminal set Ψ , and w_{ij} refers to the connectivity (or synergy) from meme m_j to m_i on solution s_i . In terms of individual learning, w_{ij} denotes the degree to which m_i 's activation favors m_j 's activation, thus providing an indication on the preferences of one type of interaction over another during the memeplex individual learning.

The workflow of the SAMRS individual learning, in which memeplex $\mathcal{M}_i$ serves as the adaptive local searcher to s_i , is illustrated in Alg. 11. Within the individual learning process,

memeplex $\mathcal{M}_i$ adaptively *online* selects and executes the memes from its terminal set $\Psi = \{m_k\}_{k=1}^\kappa$, to locally refine solution s_i . The individual learning terminates when an adaptive termination criteria based on simulated-annealing is reached.

Algorithm 11 Outline of *Memeplex Individual-Learning*($s_i, \mathcal{M}_i$)

Input: s_i, W_i

Output: s_i (solution refined), W_i (updated strategy matrix)

$\Delta f(s_i, \mathcal{M}_i) \leftarrow 0$

$\Delta T(s_i, \mathcal{M}_i) \leftarrow 0$

$t \leftarrow 0$

$m_{\pi(t)} \leftarrow \emptyset$

while TRUE **do**

Adaptively Select meme $m_k \in \Psi$ with probability $P_{\text{activate}}(m_k | m_{\pi(t)}, W_i)$
 $(\Delta f(s_i, m_k), \Delta T(s_i, m_k)) \leftarrow$ **Refine** s_i by m_k in the context of RS^3

/* Assign credit to meme pair $m_{\pi(t)} \rightarrow m_k$ based on feedback */

$w_{\pi(t)k} \leftarrow \gamma w_{\pi(t)k} + \frac{\Delta f(s_i, m_k)}{\Delta T(s_i, m_k)}$

/* Update memeplex credit */

$\Delta f(s_i, \mathcal{M}_i) += \Delta f(s_i, m_k)$

$\Delta T(s_i, \mathcal{M}_i) += \Delta T(s_i, m_k)$

/* Determine whether the individual learning should terminate */

if $\Delta f(s_i, m_k) == 0$ **then**

if $\text{rand}(0, 1) > \exp(-\frac{\Delta T(s_i, \mathcal{M}_i)}{\Delta f(s_i, \mathcal{M}_i)})$ **then**

Break

end if

end if

$t \leftarrow t + 1$

$m(t) \leftarrow m_k$

end while

The activity of $\mathcal{M}_i$ during the individual learning process is a series of adaptive memetic activations which iterate with each activation mobilizing a meme $m_j \in \mathcal{M}_i$ to locally refine solution s_i . As illustrated in Alg. 11, the individual learning process consists of two simultaneous, interacting processes, one making the matrix W_i consistent with the individual memes

and strategy of meme co-adaptation in memeplex $\mathcal{M}_i$ (evaluation), and the other making meme selection and collaboration strategy in $\mathcal{M}_i$ greedy with respect to W_i (improvement). These two processes alternate within the individual learning process, with one starting after another completes its task, until the individual learning stops, thus allowing individual meme fitness and memetic synergy to synchronize with the structure of individual solution s_i .

These two alternating processes of evaluation and improvement are detailed below:

☞ **Evaluation:** In this process, the search performances of each activated meme, in terms of $\{\Delta f(s_i, m_k), \Delta T(s_i, m_k)\}$ (where $\Delta f(s_i, m_k)$ and $\Delta T(s_i, m_k)$ denotes solution fitness improvement and computational cost made by m_k), are recorded and used as evaluative feedback for adjusting the individual meme's fitness and the memetic synergy between memes (i.e, entries in W_i), with respect to solution s_i . The detailed mechanism for W_i adjustment is described in what follows: after $m_{\pi(t)} = m_i$ was selected by memeplex $\mathcal{M}_i$ to refine solution s_i at individual learning step t , suppose $m_{\pi(t+1)} = m_j$ is then selected by memeplex $\mathcal{M}_i$ as the next meme to be applied at individual step $t+1$, after m_j refines the solution s_i , the memetic synergy $w_{\pi(t)\pi(t+1)} = w_{ij} \in W_i$ is updated according to the equation given below:

$$w_{ij} = \gamma w_{ij} + r_j \quad (\text{Eq. 5.14})$$

$$= \gamma w_{ij} + \frac{\Delta C_j}{\Delta T_j} \quad (\text{Eq. 5.15})$$

☞ **Improvement:** In this process, the next meme m_k to refine solution s_i is selected with selection probability $P_{\text{activate}}(m_k|m_{\pi(t)})$ given by:

$$P_{\text{activate}}(m_k|m_{\pi(t)}, W_i) = \begin{cases} \frac{w_{\pi(t)k}}{\sum_{\mu=1}^{\kappa} w_{\pi(t)\mu}} & m_{\pi(t)} \neq \emptyset \\ \frac{w_{kk}}{\sum_{\mu=1}^{\kappa} w_{\mu\mu}} & \text{otherwise} \end{cases} \quad (\text{Eq. 5.16})$$

where $m_{\pi(t)}$ denotes the meme selected at previous individual learning step t , $w_{\pi(t)k}$ denotes the preference of selecting m_k given $m_{\pi(t)}$. Eq. 5.16 allows memeplex adaptively select memes with bias towards high meme fitness and synergy in W_i .

For CAMA, the meme activation sequence $m_{\pi(1)} \rightarrow m_{\pi(2)} \rightarrow \dots \rightarrow m_{\pi(\varkappa)}$ is assembled *before* an individual learning starts, and remains *fixed* during the individual learning. On the other hand, for SAMRS individual learning, the meme activation sequence is unknown before the individual learning starts. Instead, a meme $m_k \in \Psi$ is probabilistically selected *online* based on W_i , to refine solution at each individual learning step. Furthermore, as W_i keeps updated at each individual learning step, the meme activation sequence dynamically changes as the individual learning processes. The memeplex individual learning discussed thus allows neighborhood search move to be adaptively varied and evaluated at each individual learning iteration step. In the end, while CAMA and SAMRS both search the space of memes, for the memeplex in SAMRS, self-adaptation at each individual learning step takes advantage of information available from the RS^3 fitness sampling during the course of individual learning.

5.3.4 Gene-Meme coevolution

In this section, the gene-meme coevolution in SAMRS is described. In the evolutionary phase, the genetic solutions and memeplexes undergo two simultaneous, interacting processes of genetic and memetic evolution following the principle of Universal Darwinism [40, 122] of transmission, variation and selection. During the SAMRS's evolutionary cycles, the relationship of genetic solutions and memeplex can manifest in the form of cooperative coevolution, which allows SAMRS to focus individual learning activity in regions where it can help the most.

In the VRPSD with safety stock, this gene-meme coevolutionary mechanism allows the behavior of the memeplex $\mathcal{M}_i$ to further self-adapt to the routing design of s_i and the value of safety stock level k_i , which is depicted in Fig. 5.3.

Figure 5.3: Gene-Meme coevolution Cycle among $s_i, \mathcal{M}_i$ in $\text{Pop}(g)$

5.3.4.1 Genetic and Memetic Recombination

The genetic recombination between solutions s_i and s_j consists of two separate mechanisms, i.e., crossover between $\text{route}(s_i)$ and $\text{route}(s_j)$ and crossover between k_i and k_j .

For crossover between $\text{route}(s_i)$ and $\text{route}(s_j)$, the genetic crossover operator as described in Section 3.4.1 is used. In this genetic crossover operator, the routes in child solution s_c are generated from partial routes of parent solutions s_i, s_j . In particular, taking the depot node e_0 as the axis center, a bisecting axis ψ of random angle θ is generated to slice the VRP graph into two separate subgraphs containing partial routes, H_+ and H_- , and the child solution s_c is a fusion of partial routes in s_i and s_j .

For crossover between k_i and k_j , the safety level k_c in child solution s_c is generated based on 1-dimensional linear interpolation between parent safety stock levels k_i and k_j , and given as:

$$k_c = k_i + t \cdot (k_j - k_i), t \in (0, 1) \quad (\text{Eq. 5.17})$$

For the memetic recombination between $\mathcal{M}_i$ and $\mathcal{M}_j$, the memeplex $\mathcal{M}_c$ in child solution s_c is generated from the parent memeplexes $\mathcal{M}_i$ and $\mathcal{M}_j$ by random exchange of corresponding

weight elements between weight matrices W_i and W_j . Graphically, this can be depicted by Fig. 5.4.

Figure 5.4: Memetic Recombination between M_i and M_j (top: memotype bottom: sociotype)

5.3.4.2 Genetic and Memetic Mutation

The genetic mutation process for solution s_i composes of 2 separate mechanisms, i.e., mutation of $\text{route}(s_i)$ and mutation of k_i . For mutation in $\text{route}(s_i)$, the mutation operator as described in Section 3.4.1.2 is used. In this genetic mutation operator, one of the four mutation heuristics, namely *swap*, *inversion*, *insertion* and *displacement* is selected randomly to apply [27]. For mutation of k_i , a small perturbation δ drawing from a Normal distribution is added to k_i , which is given by:

$$k'_i = k_i + \delta \quad (\text{Eq. 5.18})$$

For the memetic mutation of $\mathcal{M}_i$, a small set of weights $\{w_{ij}\}$ is randomly selected from W_i , and for each selected w_{ij} , a small perturbation δ_{ij} drawing from a Normal distribution is added. Again, relatively high mutation rate is applied with a small population size in SAMRS for convergence reliability and velocity as well as population diversity for combinatorially explosive problems such as VRPSD. The memetic crossover and memetic mutation use simple mechanism of performing crossover between two memeplexes. They are analogous to the multiple C point crossover and mutation in genetic algorithm. The objective is to produce new memeplexes which have weights inherited from fitter memeplexes already obtained. The new memeplexes give rise to new program structure of the refinement operator associated with the genetic solution. However, this is not the only way to perform memetic crossover and mutation, there can be many different ways to crossover two memeplexes, in ways analogous to the crossover operators between two continuous solution vectors.

5.3.5 Genetic Representation Repair

As aforementioned, for VRPSD, under the stochastic behaviors of the customer demands, information pertaining to the true demand of a customer is only known at each customer stop, hence it is possible for routes in solution s_i to turn infeasible as a result of capacity constraint violations. The repair of genetic solution s_i is thus necessary before fitness evaluation to ensure s_i as a feasible solution in the context of sampled customer demand distribution. The detailed implementation of the genetic representation repair is described in Section 5 of Chapter 3.

5.3.6 Workflow of SAMRS

The basic steps of the SAMRS Memetic Framework are outlined in Alg. 12. The algorithm starts with the initialization of a population $\text{Pop}(g = 0)$. Each $agt_i = \{s_i, \mathcal{M}_i\} \in \text{Pop}(g)$ is made up of both genetic and memetic materials. For the VRPSD, the genetic material s_i is composed of the route design $\{\tau_1, \tau_2, \dots, \tau_k, \dots, \tau_K\}$ and the associated safety stock level k_i , while

the memetic material refers to memes or local searches defined in the memeplex $\mathcal{M}_i$, which self-adapts along with the associated individual solution s_i in the individual learning and gene-meme coevolution. At each generation g , a population of offspring $\text{Pop}(g)$ is generated using evolutionary search, where both genetic and memetic transmission, variation, and selection processes take place through reproduction operations involving recombination and mutation, and subsequently memeplex individual learning and RS^3 fitness evaluation. The best solution s_{opt} of a population is then identified according to fitness $f(s_i)$. At this decision point, the evaluation of expected cost $C_{VRPSD}(s_{opt})$ is performed using Monte Carlo simulation, which is pitted against $C_{VRPSD}(s_g)$ for updating of the global best solution s_g . This workflow of memetic search proceeds until the stopping condition is reached (e.g., the maximum number of fitness evaluations budget allocated).

Algorithm 12 Outline of SAMRS Workflow

Input: Problem Instance**Output:** Global best solution s_g

Create an initial population $\text{Pop}(g = 0) = \{agt_1, agt_2, \dots, agt_i, \dots, agt_M\}, agt_i = \{s_i, \mathcal{M}_i\}$

while stopping conditions are not satisfied **do**

$g \leftarrow g + 1$

 /* Gene-meme coevolution */

Genetic Crossover on $\{\text{route}(s_i), k_i\}, \{\text{route}(s_j), k_j\}$ with probability p_c

Genetic Mutation on $\{\text{route}(s_i), k_i\}$ with probability p_m

Memetic Crossover on $\mathcal{M}_i, \mathcal{M}_j$ with probability p_c

Memetic Mutation on $\mathcal{M}_i$ with probability p_m

 /* Evaluate $f(s_i), \forall s_i \in \text{Pop}(g)$ */

Sample instance P_i by parameter perturbation on the original VRPSD instance

Genetic Repair s_i for infeasible routes

Fitness Evaluation: $f(s_i)$ in the context of P_i

 /* Memeplex Individual Learning */

for $agt_i = \{s_i, \mathcal{M}_i\} \in \text{Pop}(g)$ **do**

while TRUE **do**

 /* Adaptively make $\mathcal{M}_i$ select a meme and refine a solution s_i */

 /* Assign credit to meme pair in W_i based on feedback */

 /* Update memeplex credit */

 /* Determine whether the individual learning should terminate */

end while

end for

 /* Update global best solution s_g */

Select $s_{opt} = \max_{f(s_i)} \{s_i | s_i \in \text{Pop}(g)\}$

Evaluate $C_{VRPSD}(s_{opt})$ by performing Monte Carlo simulation on s_{opt} based on $PDF(\mu_i, \sigma_i^2)$

$s_g \leftarrow s_{opt}$ if $C_{VRPSD}(s_{opt}) < C_{VRPSD}(s_g)$

end while

5.4 Empirical Study

In this section, an empirical study on the SAMRS for solving VRPSD is presented. In order to analyze the proposed SAMRS, the algorithm was pitted against, two algorithms, namely *CVRP* and *OBS*, which denote recently proposed state-of-the-art methodology [85] as the baseline algorithms, for the discovery of robust and reliable solutions in safety stock based VRPSD. The VRPSD benchmarks used in this empirical study are the same as that in [85], which were generalized from a set of 55 classical CVRP instances [172]. These problem instances include 14 random instances from [8] set A with sizes ranging from 32 to 80, 13 clustered instances from Augerat set B with sizes ranging from 31 to 78, 16 instances from Augerat set P with sizes ranging from 19 to 101, 11 instances from sets E, M and F by [31] and [49]. They differ in terms of size (i.e., number of customers), customer distribution, symmetric structure, and vehicle capacity. All the node coordinates and vehicle capacities in the CVRP instances are kept consistent to [85], with the deterministic customer demands $\{d_i\}$ changed to stochastic demands $\{demand(i)\}$ with expectation $E[demand(i)] = d_i$ and variance $Var[demand(i)] = \alpha \cdot d_i$.

5.4.1 Experimental Setup

All results presented in this section are the summaries obtained from 30 independent runs for each VRPSD instance under a consistent experimental setup. Each run continues until a maximum of 50,000 fitness evaluations is reached. In each run, the search settings of the memetic algorithm used for solving the benchmark problems are defined as follows: population size of 30, crossover probability of $p_c = 0.5$, mutation rate of $p_m = 0.5$, roulette-wheel selection scheme, and an elite size of 2. The settings are chosen to encourage greater exploration and hence diversity in the population during the search.

Statistical results with respect to the following metrics are then reported:

BKS	- denotes the fixed cost of the best known solution $CVRP$ from the corresponding CVRP instance
C_0	- denotes the expected cost of the best known solution s_{bks} on the VRPSD instance
C_1	- denotes the expected cost for the best solution obtained by OBS on the VRPSD instance
C_2	- denotes the expected cost for the best solution obtained by $SAMRS$ on the VRPSD instance
$\overline{C_2}$	- denotes the average of the converged solution's expected cost on the VRPSD instance over $NR = 30$ independent runs. $\overline{C_2} = \frac{\sum runs C_{run}}{NR}$
R_0	- denotes the route reliability level associated with the solution having expected cost C_0
R_1	- denotes the route reliability level associated with the solution having expected cost C_1 in OBS
R_2	- denotes the route reliability level associated with the solution having expected cost C_2 in $SAMRS$
$\overline{R_2}$	- denotes the average of the converged solution's reliability on the VRPSD instance over $NR = 30$ independent. $\overline{R_2} = \frac{\sum runs R_{run}}{NR}$
K_1	- denotes number of routes in solution having expected cost C_1 in OBS
K_2	- denotes number of routes in solution having expected cost C_2 in $SAMRS$
k_1	- denotes safety stock level in solution having expected cost C_1 in OBS
k_2	- denotes safety stock level in solution having expected cost C_2 in $SAMRS$
Gap_0	- denotes the average gap between C_0 and BKS over 55 instances. $Gap_0 = \frac{C_0 - BKS}{BKS} \times 100\%$
Gap_1	- denotes the average gap between C_1 and BKS over 55 instances. $Gap_1 = \frac{C_1 - BKS}{BKS} \times 100\%$
Gap_2	- denotes the average gap between C_2 and BKS over 55 instances. $Gap_2 = \frac{C_2 - BKS}{BKS} \times 100\%$
$\overline{Gap_2}$	- denotes the average gap between C_2 and BKS over 55 instances. $\overline{Gap_2} = \frac{\overline{C_2} - BKS}{BKS} \times 100\%$

5.4.2 Results and Discussion

Tables 5.1, 5.2, 5.3 show the complete results obtained for all 55 VRPSD instances. The results of $CVRP$ and OBS are obtained directly from [85]. Each of these tables corresponds to one of the three uncertainty scenarios (low-variance, medium-variance and high-variance), which were previously described in Section 5.4.1. In the tables, the instance name is given by “[SetId]-n[Size]-k[VehicleCount]”, where “[SetId]” denotes the id of the benchmark set (e.g., A, B, E, P, etc.), “[Size]” denotes the problem size (i.e., the total number of customers served), and “[VehicleCount]” denotes the vehicle count specified in the original classical instance. The values, highlighted in bold, in the tables, report the best results arrived on a particular problem instance by all the algorithms under comparison.

Table 5.1: Results of the 55 VRPSD instances in a low-variance scenario $\alpha = 0.05$ (The best performing algorithm of the respective VRPSD instance is highlighted in Bold)

#	Instance	CVRP			OBS				SAMRS					
		BKS	C_0	R_0	C_1	k_1	K_1	R_1	C_2	k_2	K_2	R_2	C_2	R_2
1	A-n32-k5	787.08	828.24	0.55	797.74	0.96	5	1.00	796.16	0.82	5	1.00	796.20	0.94
2	A-n33-k5	662.11	689.28	0.60	678.14	0.97	5	0.90	675.71	0.93	6	1.00	677.12	0.97
3	A-n33-k6	742.69	770.58	0.57	762.02	0.98	6	0.67	756.07	0.95	7	0.90	756.94	0.96
4	A-n37-k5	672.47	687.68	0.78	685.36	0.96	5	0.93	677.99	0.84	5	1.00	678.67	0.98
5	A-n38-k5	733.95	773.46	0.41	754.04	0.95	6	0.99	748.77	0.92	6	1.00	749.38	0.97
6	A-n39-k6	833.21	940.65	0.16	839.12	0.96	6	0.96	835.83	0.88	7	1.00	841.18	0.96
7	A-n45-k6	944.88	1076.45	0.10	975.12	0.96	7	0.93	969.01	0.99	7	0.95	969.42	0.97
8	A-n45-k7	1146.77	1299.53	0.20	1169.85	0.95	7	0.95	1165.63	0.85	8	1.00	1185.19	0.94
9	A-n55-k9	1074.46	1205.57	0.04	1106.31	0.96	9	0.87	1115.23	0.82	10	1.00	1119.16	0.95
10	A-n60-k9	1355.80	1517.79	0.12	1383.90	0.95	9	0.94	1377.54	0.94	10	0.93	1385.09	0.96
11	A-n61-k9	1039.08	1145.39	0.09	1073.99	0.97	10	0.75	1068.74	0.95	11	1.00	1070.97	0.98
12	A-n63-k9	1622.15	1884.98	0.08	1682.37	0.95	10	0.95	1667.58	0.80	10	1.00	1699.61	0.97
13	A-n65-k9	1181.69	1352.64	0.06	1241.51	0.96	10	0.99	1234.73	0.83	11	1.00	1242.54	0.91
14	A-n80-k10	1766.50	2028.28	0.08	1828.40	0.96	10	0.86	1864.78	0.92	12	1.00	1893.68	0.97
15	B-n31-k5	676.09	726.15	0.50	682.65	0.95	5	0.99	678.28	0.96	5	1.00	678.49	1.00
16	B-n35-k5	956.29	1068.48	0.45	986.58	0.94	5	1.00	990.43	0.94	5	1.00	995.31	0.99
17	B-n39-k5	553.16	589.74	0.49	564.47	0.96	5	0.96	561.70	0.98	5	1.00	561.75	0.98
18	B-n41-k6	834.92	912.97	0.23	858.91	0.95	7	0.99	855.64	0.98	8	1.00	878.99	0.95
19	B-n45-k5	754.22	801.66	0.23	764.27	0.94	6	1.00	761.53	0.93	6	1.00	762.34	0.99
20	B-n50-k7	744.23	800.04	0.31	752.54	0.95	7	1.00	748.64	0.91	8	1.00	749.02	0.99
21	B-n52-k7	749.96	826.93	0.33	762.26	0.95	7	0.99	758.22	0.91	8	1.00	760.11	0.99
22	B-n56-k7	712.92	830.88	0.17	739.23	0.95	7	0.98	730.68	0.97	8	1.00	732.93	0.99
23	B-n57-k9	1602.28	1922.49	0.06	1626.89	0.95	9	0.98	1634.39	0.96	9	1.00	1647.15	0.99
24	B-n64-k9	868.31	1070.90	0.02	903.31	0.95	10	0.98	908.05	0.91	10	0.85	910.97	0.92
25	B-n67-k10	1039.36	1207.88	0.08	1096.78	0.96	10	0.92	1092.30	0.93	12	1.00	1098.46	0.98
26	B-n68-k9	1276.20	1525.45	0.04	1311.58	0.94	9	0.99	1313.42	1.00	10	1.00	1329.13	0.97
27	B-n78-k10	1227.90	1370.36	0.11	1299.00	0.97	10	0.76	1300.69	0.98	11	1.00	1304.64	0.97
28	E-n22-k4	375.28	375.28	1.00	375.28	1.00	4	1.00	375.00	0.99	4	1.00	380.00	1.00
29	E-n30-k3	505.01	505.01	1.00	505.01	1.00	4	1.00	503.00	1.00	4	1.00	503.80	1.00
30	E-n33-k4	837.67	838.80	0.99	838.80	1.00	4	1.00	835.70	0.95	5	1.00	839.42	1.00
31	E-n51-k5	524.61	539.21	0.27	537.11	0.98	5	0.65	533.02	0.90	6	1.00	537.89	0.96
32	E-n76-k10	835.26	905.54	0.07	861.77	0.98	11	0.70	858.64	0.86	13	0.94	867.81	0.93
33	E-n76-k14	1024.40	1121.13	0.04	1056.36	0.96	15	0.90	1068.57	0.95	16	1.00	1078.48	0.96
34	E-n76-k7	687.60	703.46	0.58	695.49	0.97	7	1.00	689.01	0.97	7	1.00	691.05	0.99
35	F-n135-k7	1164.73	1263.53	0.46	1191.26	0.99	7	0.99	1189.82	0.92	7	1.00	1203.69	0.99
36	F-n45-k4	723.54	757.83	0.37	729.97	0.96	5	1.00	729.97	0.95	5	1.00	734.59	1.00
37	F-n72-k4	241.97	247.76	0.70	247.29	0.99	4	1.00	244.92	0.86	5	1.00	250.38	1.00
38	M-n101-k10	819.56	933.10	0.07	866.83	0.95	10	0.99	825.25	0.87	8	1.00	833.31	0.98
39	M-n121-k7	1043.88	1101.50	0.08	1074.85	0.98	8	0.83	1063.82	0.97	8	1.00	1067.81	0.98
40	P-n101-k4	691.29	693.52	0.96	693.52	0.98	4	0.96	681.03	0.94	5	1.00	688.55	1.00
41	P-n19-k2	212.66	217.48	0.85	216.96	0.98	2	0.85	209.95	0.82	3	1.00	210.07	0.84
42	P-n20-k2	217.42	233.06	0.65	220.89	0.99	2	0.93	218.86	0.86	3	1.00	219.06	0.84
43	P-n22-k2	217.85	226.34	0.65	224.13	0.97	2	0.96	221.92	0.96	2	1.00	222.43	0.95
44	P-n22-k8	588.79	619.74	0.50	589.39	0.97	9	1.00	590.00	0.96	9	1.00	602.30	1.00
45	P-n40-k5	461.73	466.25	0.77	464.51	0.97	5	1.00	461.00	0.84	6	1.00	461.41	1.00
46	P-n50-k10	699.56	753.27	0.16	721.91	0.96	10	0.90	725.36	1.00	11	0.99	727.86	0.97
47	P-n50-k8	632.69	697.56	0.10	646.65	0.97	9	0.91	643.95	0.86	9	1.00	647.11	0.98
48	P-n51-k10	741.50	828.90	0.05	761.62	0.96	11	0.90	757.75	0.94	12	1.00	762.45	0.94
49	P-n55-k15	944.56	1033.66	0.04	991.85	0.95	17	0.79	990.06	0.99	17	0.97	992.44	0.97
50	P-n55-k7	570.27	588.59	0.39	578.61	0.95	7	1.00	576.00	0.85	8	1.00	577.75	1.00
51	P-n60-k10	748.07	790.46	0.20	761.49	0.97	10	0.85	763.78	0.82	10	0.96	773.06	0.98
52	P-n65-k10	795.66	854.65	0.13	811.83	0.97	10	0.92	814.37	0.85	10	0.98	823.53	0.98
53	P-n70-k10	829.93	906.87	0.05	853.83	0.96	11	0.94	846.13	0.83	11	0.99	858.26	0.98
54	P-n76-k4	598.19	611.81	0.49	609.31	0.98	5	0.97	599.56	0.87	5	0.98	612.84	0.98
55	P-n76-k5	633.32	651.51	0.34	648.86	0.98	5	0.85	643.42	0.86	6	1.00	648.22	0.99

Table 5.2: Results of the 55 VRPSD instances in a medium-variance scenario $\alpha = 0.25$ (The best performing algorithm of the respective VRPSD instance is highlighted in Bold)

#	Instance	CVRP			OBS				SAMRS					
		BKS	C_0	R_0	C_1	k_1	K_1	R_1	C_2	k_2	K_2	R_2	C_2	R_2
1	A-n32-k5	787.08	867.93	0.29	815.78	0.92	5	0.86	812.35	0.80	5	1.00	814.64	0.97
2	A-n33-k5	662.11	726.04	0.30	693.09	0.95	5	0.81	689.80	0.98	6	1.00	690.90	0.93
3	A-n33-k6	742.69	792.49	0.29	775.19	0.95	7	0.89	766.79	0.97	6	0.94	772.59	0.89
4	A-n37-k5	672.47	707.31	0.53	701.38	0.93	5	0.79	691.36	0.94	6	1.00	695.86	0.91
5	A-n38-k5	733.95	801.86	0.19	762.87	0.93	6	0.93	753.48	0.90	6	1.00	759.50	0.95
6	A-n39-k6	833.21	963.15	0.08	858.78	0.92	6	0.92	850.76	0.92	7	1.00	858.54	0.97
7	A-n45-k6	944.88	1115.53	0.04	990.74	0.94	7	0.78	988.79	0.99	7	1.00	991.99	0.97
8	A-n45-k7	1146.77	1375.16	0.09	1208.11	0.93	7	0.72	1215.24	0.97	8	1.00	1220.51	0.90
9	A-n55-k9	1074.46	1214.44	0.06	1137.69	0.94	10	0.70	1137.90	0.91	11	1.00	1145.68	0.93
10	A-n60-k9	1355.80	1576.73	0.07	1452.36	0.93	10	0.74	1454.83	0.96	10	0.96	1466.67	0.90
11	A-n61-k9	1039.08	1193.77	0.03	1105.31	0.92	10	0.77	1096.94	0.86	11	1.00	1105.48	0.97
12	A-n63-k9	1622.15	2002.01	0.03	1769.05	0.93	10	0.70	1781.07	0.95	10	0.98	1794.66	0.90
13	A-n65-k9	1181.69	1403.74	0.02	1274.47	0.96	10	0.62	1268.44	0.87	11	1.00	1287.42	0.95
14	A-n80-k10	1766.50	2126.74	0.03	1923.30	0.89	11	0.91	1940.02	0.94	11	1.00	1967.24	0.92
15	B-n31-k5	676.09	753.82	0.39	695.57	0.91	5	0.93	690.45	0.99	5	0.65	691.87	0.85
16	B-n35-k5	956.29	1115.03	0.33	992.01	0.89	5	0.97	999.71	0.85	5	1.00	1004.72	0.93
17	B-n39-k5	553.16	608.82	0.37	579.60	0.94	5	0.78	572.39	0.89	5	1.00	573.01	0.99
18	B-n41-k6	834.92	969.74	0.13	884.28	0.92	7	0.86	891.41	0.94	7	1.00	903.97	0.99
19	B-n45-k5	754.22	827.48	0.13	767.74	0.90	6	0.94	764.68	0.94	6	0.99	766.08	0.95
20	B-n50-k7	744.23	827.32	0.20	761.54	0.89	7	0.94	759.25	0.89	7	1.00	760.31	0.92
21	B-n52-k7	749.96	848.27	0.25	774.01	0.89	7	0.97	771.54	0.89	7	1.00	774.09	0.94
22	B-n56-k7	712.92	856.15	0.11	755.90	0.89	7	0.94	747.66	0.95	8	1.00	752.47	0.93
23	B-n57-k9	1602.28	2003.64	0.03	1686.40	0.91	9	0.84	1711.55	0.97	9	0.93	1725.40	0.97
24	B-n64-k9	868.31	1086.15	0.01	929.61	0.92	10	0.81	926.23	0.95	10	0.98	933.57	1.00
25	B-n67-k10	1039.36	1277.39	0.04	1121.49	0.93	11	0.85	1115.18	0.94	11	1.00	1121.42	0.97
26	B-n68-k9	1276.20	1634.31	0.01	1381.21	0.95	9	0.44	1396.91	0.93	10	1.00	1416.50	0.94
27	B-n78-k10	1227.90	1522.01	0.01	1333.02	0.91	11	0.85	1329.79	0.96	11	1.00	1334.93	0.93
28	E-n22-k4	375.28	375.49	1.00	375.49	1.00	4	1.00	375.06	0.94	5	1.00	375.08	1.00
29	E-n30-k3	505.01	505.01	1.00	505.01	1.00	4	1.00	503.00	0.96	4	1.00	503.50	1.00
30	E-n33-k4	837.67	858.91	0.88	844.08	0.99	4	0.99	839.01	0.82	5	1.00	840.13	0.99
31	E-n51-k5	524.61	551.21	0.17	548.63	0.97	6	0.60	544.45	0.89	6	1.00	546.56	0.95
32	E-n76-k10	835.26	935.03	0.03	873.81	0.92	11	0.89	865.65	0.98	11	0.85	880.70	0.93
33	E-n76-k14	1024.40	1169.79	0.02	1087.01	0.93	16	0.64	1091.48	0.85	18	1.00	1106.59	0.94
34	E-n76-k7	687.60	717.42	0.37	700.55	0.96	7	0.80	691.54	0.98	7	1.00	697.06	0.95
35	F-n135-k7	1164.73	1271.72	0.43	1200.91	0.97	7	1.00	1194.11	0.99	7	0.98	1208.27	0.99
36	F-n45-k4	723.54	769.79	0.34	729.97	0.96	5	1.00	729.97	0.97	5	1.00	729.98	1.00
37	F-n72-k4	241.97	249.92	0.56	247.29	0.99	4	1.00	247.57	0.99	5	1.00	251.57	1.00
38	M-n101-k10	819.56	939.56	0.06	866.16	0.98	10	0.62	829.58	0.99	9	1.00	845.72	0.93
39	M-n121-k7	1043.88	1178.76	0.06	1140.95	0.96	8	0.60	1128.28	0.97	8	1.00	1149.91	0.87
40	P-n101-k4	691.29	698.25	0.74	695.92	0.96	4	0.95	682.55	0.99	5	1.00	689.62	0.97
41	P-n19-k2	212.66	228.35	0.59	222.37	0.94	3	0.96	208.97	0.94	3	1.00	209.20	0.91
42	P-n20-k2	217.42	238.83	0.53	232.49	0.97	2	0.63	223.00	0.90	3	1.00	223.17	0.91
43	P-n22-k2	217.85	229.08	0.54	229.08	1.00	2	0.54	225.86	0.97	3	1.00	226.00	0.93
44	P-n22-k8	588.79	619.77	0.50	589.41	0.98	9	1.00	590.00	0.87	9	1.00	591.01	1.00
45	P-n40-k5	461.73	470.99	0.60	466.92	0.96	5	0.90	463.23	0.91	6	0.96	463.44	0.98
46	P-n50-k10	699.56	783.61	0.05	736.71	0.92	11	0.82	734.01	0.82	12	0.97	745.36	0.98
47	P-n50-k8	632.69	711.47	0.04	655.67	0.92	9	0.89	650.47	0.85	10	1.00	655.37	0.97
48	P-n51-k10	741.50	851.78	0.02	785.64	0.96	11	0.39	786.72	0.97	12	1.00	790.58	0.94
49	P-n55-k15	944.56	1114.75	0.00	1032.11	0.93	18	0.55	1028.16	0.96	18	1.00	1035.99	0.98
50	P-n55-k7	570.27	608.59	0.18	582.94	0.93	7	0.90	578.95	0.98	7	1.00	583.85	0.96
51	P-n60-k10	748.07	821.07	0.07	790.12	0.92	11	0.93	792.43	0.96	11	0.97	796.17	0.95
52	P-n65-k10	795.66	854.79	0.11	835.04	0.95	10	0.42	834.45	0.93	11	1.00	844.83	0.91
53	P-n70-k10	829.93	946.00	0.02	868.20	0.93	11	0.90	858.38	0.85	12	0.97	873.34	0.96
54	P-n76-k4	598.19	616.10	0.33	616.07	0.98	5	0.69	606.76	0.84	5	1.00	613.93	1.00
55	P-n76-k5	633.32	656.29	0.20	651.26	0.96	6	0.82	649.20	0.99	6	0.99	652.85	0.96

Table 5.3: Results of the 55 VRPSD instances in a high-variance scenario $\alpha = 0.75$ (The best performing algorithm of the respective VRPSD instance is highlighted in Bold)

#	Instance	CVRP			OBS				SAMRS					
		BKS	C_0	R_0	C_1	k_1	K_1	R_1	C_2	k_2	K_2	R_2	C_2	R_2
1	A-n32-k5	787.08	889.99	0.23	853.63	0.92	5	0.55	850.47	0.81	5	0.96	852.24	0.81
2	A-n33-k5	662.11	751.24	0.20	715.18	0.89	6	0.77	706.32	0.95	5	0.88	708.92	0.79
3	A-n33-k6	742.69	814.24	0.18	803.96	0.95	7	0.58	791.32	0.94	6	0.86	800.28	0.80
4	A-n37-k5	672.47	732.29	0.36	719.09	0.85	5	0.91	707.05	0.82	5	1.00	711.89	0.97
5	A-n38-k5	733.95	820.56	0.13	777.18	0.90	6	0.86	770.99	0.93	6	1.00	773.33	0.95
6	A-n39-k6	833.21	969.20	0.07	890.16	0.93	6	0.60	883.49	0.91	7	1.00	890.50	0.95
7	A-n45-k6	944.88	1142.27	0.03	1026.08	0.95	7	0.44	1022.54	0.97	7	1.00	1027.54	0.91
8	A-n45-k7	1146.77	1411.09	0.06	1279.20	0.93	7	0.35	1270.77	0.91	8	1.00	1299.60	0.93
9	A-n55-k9	1074.46	1259.62	0.04	1189.56	0.87	10	0.71	1186.39	0.93	10	1.00	1201.57	0.93
10	A-n60-k9	1355.80	1631.51	0.04	1539.37	0.94	10	0.30	1545.51	0.91	10	1.00	1554.61	0.86
11	A-n61-k9	1039.08	1223.96	0.01	1154.93	0.93	10	0.34	1146.64	0.89	11	1.00	1153.28	0.87
12	A-n63-k9	1622.15	2047.45	0.01	1867.97	0.89	10	0.47	1865.45	0.91	10	0.87	1895.08	0.88
13	A-n65-k9	1181.69	1424.04	0.01	1321.27	0.91	10	0.43	1336.18	0.92	12	1.00	1341.17	0.87
14	A-n80-k10	1766.50	2207.69	0.02	1994.10	0.89	11	0.53	2031.81	0.92	11	1.00	2050.43	0.85
15	B-n31-k5	676.09	775.84	0.32	713.90	0.89	5	0.86	709.74	0.81	5	0.80	716.91	0.80
16	B-n35-k5	956.29	1145.13	0.22	1036.03	0.91	5	0.66	1040.93	0.90	5	1.00	1048.46	0.93
17	B-n39-k5	553.16	655.32	0.27	602.72	0.88	6	0.78	591.86	0.95	5	0.87	596.15	0.84
18	B-n41-k6	834.92	1008.46	0.09	929.01	0.88	7	0.86	925.95	0.98	8	1.00	929.06	0.95
19	B-n45-k5	754.22	847.67	0.09	790.20	0.92	6	0.61	781.24	0.94	6	1.00	783.51	0.91
20	B-n50-k7	744.23	853.48	0.14	790.54	0.89	7	0.61	788.15	0.96	8	1.00	791.95	0.96
21	B-n52-k7	749.96	881.49	0.16	794.33	0.89	7	0.66	794.55	0.98	7	1.00	798.92	0.83
22	B-n56-k7	712.92	886.93	0.08	774.66	0.88	8	0.80	772.63	0.92	8	0.97	778.14	0.81
23	B-n57-k9	1602.28	2059.75	0.02	1793.16	0.91	9	0.38	1829.80	0.87	10	1.00	1849.53	0.86
24	B-n64-k9	868.31	1103.81	0.01	968.50	0.84	11	0.85	961.11	0.94	11	1.00	965.55	0.96
25	B-n67-k10	1039.36	1287.63	0.02	1170.91	0.88	11	0.63	1164.81	0.88	12	1.00	1178.39	0.87
26	B-n68-k9	1276.20	1660.42	0.01	1453.30	0.90	10	0.46	1487.98	0.97	11	1.00	1505.38	0.94
27	B-n78-k10	1227.90	1582.81	0.01	1383.07	0.86	12	0.69	1379.10	0.88	11	0.91	1393.62	0.84
28	E-n22-k4	375.28	378.28	0.93	378.25	0.99	4	0.93	376.10	0.90	5	1.00	376.18	0.99
29	E-n30-k3	505.01	505.01	1.00	505.01	0.88	4	1.00	503.00	0.93	4	1.00	504.00	1.00
30	E-n33-k4	837.67	883.62	0.73	844.13	0.98	4	1.00	840.80	0.81	5	1.00	842.43	0.96
31	E-n51-k5	524.61	561.25	0.12	555.26	0.94	6	0.73	551.13	0.91	6	0.95	554.54	0.94
32	E-n76-k10	835.26	970.33	0.01	894.96	0.92	11	0.44	900.90	0.98	11	1.00	912.63	0.89
33	E-n76-k14	1024.40	1221.11	0.00	1127.04	0.89	16	0.43	1136.53	0.96	16	0.92	1153.35	0.93
34	E-n76-k7	687.60	743.71	0.09	704.48	0.93	7	0.79	698.73	0.96	7	1.00	707.16	0.97
35	F-n135-k7	1164.73	1280.75	0.40	1208.99	0.97	7	0.91	1216.47	0.98	8	1.00	1226.02	0.96
36	F-n45-k4	723.54	772.90	0.30	731.01	0.96	5	0.98	730.80	0.91	5	1.00	730.92	0.98
37	F-n72-k4	241.97	252.15	0.45	247.51	0.99	4	0.99	247.51	0.91	5	1.00	249.67	1.00
38	M-n101-k10	819.56	943.69	0.06	922.16	0.96	10	0.30	854.43	0.88	9	0.94	858.66	0.92
39	M-n121-k7	1043.88	1230.53	0.03	1218.10	0.98	8	0.14	1204.51	0.86	8	1.00	1232.85	0.84
40	P-n101-k4	691.29	703.87	0.53	699.87	0.96	4	0.78	686.26	0.99	4	0.94	694.14	0.92
41	P-n19-k2	212.66	236.63	0.46	227.41	0.95	3	0.85	211.31	0.95	3	1.00	211.52	0.83
42	P-n20-k2	217.42	244.14	0.44	239.34	0.99	2	0.48	225.25	0.83	3	1.00	225.47	0.77
43	P-n22-k2	217.85	232.30	0.47	232.30	1.00	2	0.47	228.94	0.86	3	1.00	229.17	0.78
44	P-n22-k8	588.79	620.24	0.49	591.05	0.98	9	0.97	591.39	0.97	9	1.00	591.57	0.99
45	P-n40-k5	461.73	477.88	0.47	474.21	0.95	5	0.64	470.29	0.93	5	1.00	475.38	0.93
46	P-n50-k10	699.56	806.07	0.02	763.24	0.91	11	0.44	762.67	0.90	11	0.93	774.63	0.92
47	P-n50-k8	632.69	733.49	0.02	675.00	0.93	9	0.47	671.29	0.98	9	0.84	677.02	0.85
48	P-n51-k10	741.50	876.30	0.01	822.43	0.96	11	0.14	819.07	1.00	11	1.00	822.86	0.89
49	P-n55-k15	944.56	1156.71	0.00	1087.04	0.87	19	0.52	1076.66	0.91	19	1.00	1087.28	0.91
50	P-n55-k7	570.27	625.80	0.09	593.15	0.92	7	0.79	589.22	0.94	8	1.00	599.56	0.89
51	P-n60-k10	748.07	869.55	0.02	809.56	0.92	11	0.54	815.36	0.95	11	1.00	824.16	0.89
52	P-n65-k10	795.66	904.90	0.05	861.92	0.93	11	0.33	864.46	0.90	12	1.00	874.14	0.95
53	P-n70-k10	829.93	959.11	0.01	889.26	0.93	11	0.47	884.89	0.97	11	0.91	904.10	0.85
54	P-n76-k4	598.19	622.71	0.18	621.75	0.89	5	0.99	612.04	0.96	5	1.00	617.80	0.86
55	P-n76-k5	633.32	660.43	0.12	659.67	0.90	6	0.96	654.28	0.99	6	1.00	658.00	0.93

The route reliability level refers to the probability (i.e., R_1 and R_2) that solution routes do not suffer from failure (i.e., capacity constraint violation) in the actual distribution phase. This is an important measure besides the expected cost, since the managerial control over frequency of route failure and expected cost penalty is dependent on the route reliability level. Monte Carlo simulation for computation of expected cost and reliability level is configured in the study at 10,000 trials. Three different statistical customer demand distributions are considered, with the variance level α configured to 0.05, 0.25, and 0.75, which simulates low, medium and high demand variances, respectively.

Figure 5.5: Gaps for solution at different variance levels (CVRP-best $\equiv Gap_0$; OBS-best $\equiv Gap_1$; SAMRS-best $\equiv Gap_2$; SAMRS-avg $\equiv \bar{Gap}_2$)

5.4.2.1 Solution Quality

On solution quality, the relative performances of the algorithms in terms of the percentage gap of C_0 , C_1 , and C_2 are depicted in Fig. 5.5. As observed from the figure, *SAMRS* consistently produces the lowest gap Gap_2 when compared to Gap_0 and Gap_1 attained by *CVRP* and *OBS*, respectively. This suggests that *SAMRS* obtained the best solution with lowest expected cost for different customer demand uncertainty levels, indicating that its superiority, even at high-variance of customer demands.

Figure 5.6: Reliability indices for solution at different variance levels ($CVRP\text{-best} \equiv R_0$; $OBS\text{-best} \equiv R_1$; $SAMRS\text{-best} \equiv R_2$; $SAMRS\text{-avg} \equiv \bar{R}_2$)

It is notable from the values of K_1 and K_2 in Tables 5.1, 5.2, 5.3 that the best solution obtained by *OBS* and *SAMRS* differs in terms of the number of routes, i.e., the best found solutions obtained by *SAMRS* have larger number of routes on approximately half of the problem instances considered. Typically, a larger number of routes than those of *OBS*, although indicating higher reliability, may also lead to higher expected costs. The results presented in Tables 5.1, 5.2, 5.3, however showed otherwise. Even for problem instances in which the best found solutions obtained by *SAMRS* have larger number of routes, the corresponding expected cost of the *SAMRS* solution are observed to remain lower than those arrived by *OBS*. Notably, the superior solution quality obtained by *SAMRS* in terms of expected cost, is contributed by the enhanced search performance of *SAMRS*, particularly a result of the adaptive search intensification brought about by the memeplex based individual learning provided in the *SAMRS*.

To further analyze the performances of *SAMRS* in terms of search efficiency and solution quality, the results obtained in Tables 5.1, 5.2, 5.3 in terms of the expected cost and solution reliability level of best found solutions and average converged solutions are pitted against the best solutions obtained by *OBS* and *CVRP* at different variance levels in customer demands,

as summarized in Figs. 5.5 and 5.6.

5.4.2.2 Solution Reliability

On solution reliability, the reliability indices of the best found solutions at different variance levels are depicted in Fig. 5.6. The figure indicates that in terms of solution reliability, the best found solution obtained by *SAMRS* is consistently superior since it exhibits a much lower probability of route failure for the different levels of variance in customer demands. It is worth noting that even at high variance of customer demands ($\alpha = 0.75$), the best solution obtained by *SAMRS* remains to maintain a high average reliability level of $R_2 = 0.975$, while the reliability level of the best found solution obtained by *OBS* drops to $R_1 = 0.627$.

The higher reliability levels obtained over *OBS*, can also be partially attributed to the way the safety stock level k is chosen. In *OBS*, the optimal solution, in terms of expected cost and reliability, is picked from alternative solutions obtained via a set of pre-defined values of k . This restricts the possible choice of k value, and thus limits the possibility of finding improved solution for which the k value exists outside the pre-defined set. In contrast, *SAMRS* models k as a continuous variable that is self-adapted along with the evolutionary search so as to enhance convergence to optimal solution.

5.4.2.3 Search Robustness

To investigate the robustness of the proposed *SAMRS*, the $\overline{C_2}$ and $\overline{R_2}$ for each VRPSD instance are also computed. $\overline{C_2}, \overline{R_2}$ denote the sampled means on expected cost and reliability of the solutions obtained by *SAMRS* across $NR = 30$ independent runs, respectively. The results in terms of the $\overline{C_2}$ and $\overline{R_2}$ are then pitted against those obtained in *OBS* and *CVRP*, at different uncertainty levels of customer demands.

The relative performances of the algorithms in terms of percentage gaps of $C_0, C_1, \overline{C_2}$ are depicted in Fig. 5.5. As illustrated by the histogram of average gaps, there is virtually no significant difference in values between the sample mean $\overline{C_2}$ obtained by *SAMRS* and the

best solution C_1 obtained by *OBS*, for the different uncertainty levels. This thus verifies the robustness of *SAMRS* in terms of solution quality.

The relative performances of the algorithms in terms of R_0 , R_1 , $\overline{R_2}$ are depicted in Fig. 5.6. As illustrated by the histogram of average gaps, while *CVRP* and *OBS* drop significantly in reliability as the variance level increase, *SAMRS* has been able to maintain highly reliable solutions across all the independent runs at low probability of route failure for the different variance levels of customer demands. Even at high variance of customer demands ($\alpha = 0.75$), the solutions obtained by *SAMRS* maintain high average reliability level of $\overline{R_2} = 0.899$, while the reliability level of the best solution obtained by *OBS* slides to $R_1 = 0.627$.

As illustrated in Figs. 5.5 and 5.6, *SAMRS* also exhibits high robustness in terms of the expected cost and reliability on the best solutions over 30 independent runs, with an average expected cost that matches well the expected cost of the best solution obtained by *OBS*, and an average reliability that is superior to that of the best solution obtained by *OBS*.

5.5 Conclusion

In this chapter, a novel Stochastic-Adaptive Memeplex Robust Search(*SAMRS*) has been introduced. *SAMRS* is composed of a memeplex individual learning, a robust solution search scheme, and a gene-meme co-evolution model. The memeplex representation bridges the two adaptive mechanisms in *SAMRS*, namely the meme coordination within the memeplex individual learning and the meme evolution within the gene-meme co-evolution model, thus efficiently managing individual learning phase for robust solution search. The robust solution search scheme in *SAMRS* directs the memetic search towards robust solutions via the Baldwinian mode of inheritance by inducing perturbations into the phenotypic features of the solution, and also reduces the computational complexity involved in the fitness evaluation by serving as an approximation to the computationally intensive Monte Carlo simulation via its implicit averaging effect. Empirical results obtained on the VRPSD benchmarks demonstrated

that SAMRS produced higher quality solutions in terms of not only the expected cost but also the solution reliability. In addition, SAMRS achieved robustness in algorithm performance under different customer demand variance level. In particular, while the compared algorithm OBS drops its solution reliability to 0.627 at high demand variance level, solutions obtained by *SAMRS* maintain a high average reliability of around 0.9 over the 30 independent runs. In summary, the SAMRS is shown to be an efficabile means of generating high-quality robust and reliable solutions in safety stock based VRPSD.

Chapter 6

Concluding Remarks

The field of memetic computation has come a long way since its inception few years ago. It has opened up many possibilities of research, in particular issues pertaining to how knowledge or information captured as units of representation known as memes are managed in the context of problem-solving. These issues are particularly significant when one begins to look at problems that are uncertain and dynamic, problems where no a priori information is known about in the solution landscape. In this chapter, we summarize the contributions of this thesis and outline some possibilities of future research issues to advance the field of memetic computation.

6.1 Research Contributions

The focus of this research is to develop *meme-centric* frameworks that harness the co-adaptive nature of memes as units of domain information housed in *multi-memes* and *memeplexes* for the purpose of effective problem-solving. By encouraging productive symbiosis among memes, the proposed *simple hybrid* and *adaptive hybrid*, which take multi-memes and memeplexes as focal point of interest in the context of memetic computation has efficaciously improved the memetic search convergence reliability, solution quality, as well as solution robustness in optimization problems such as CVRP and VRPSD. This was achieved by the effective use of memeplex representation, learning and adaptation mechanism, gene-meme coevolution model, and robust search scheme within the various hybrid forms of memetic computation.

The primary contribution of this dissertation work is summarized as follows:

- A comprehensive multi-facet survey of recent research in memetic computation was studied in Chapter 2, covering a plethora of potentially rich meme-inspired computing methodologies and frameworks. The survey helps to understand the field of memetic computation as well as the current state-of-the-art memetic approaches, it serves to facilitate the knowledge base to explore the research works in subsequent chapters.
- A simple hybrid with multi-memes individual learning was presented in Chapter 3 to reduce the inherent risk associated with choosing single meme individual learning in memetic computation for solving CVRP. The proposed memetic hybrid invests local search time budgets among multiple memes, each having different search biases, during individual learning, to reduce risks of failing on problems from different classes as well as renders finding high-quality solutions more likely.
- An adaptive hybrid with the conceptual modeling of memeplexes was explored in Chapter 4 as a self-configuring methodology capable of learning and adapting multi-memes individual learning to the given problem of interests while the search progresses online. The connection topology network of the memeplex representation, credit assignment criteria for evaluating individual memes and meme co-adaptation, as well as the role of emergent memeplexes in the individual learning process were explored, with its application for solving CVRPs of diverse characteristics.
- A memeplex robust search scheme which seeks to achieve greater level of adaptivity in problem environments with uncertainties was proposed and studied for VRPSD in Chapter 5. Via *Robust Solution Search Scheme* (RS^3), memeplex individual learning and gene-meme co-evolutionary model, the approach was demonstrated to be highly efficient and effective problem solver in finding robust and reliable solutions that are less

sensitive to stochastic behaviors of customer demands and have low probability of route failures in VRPSD.

6.2 Future Work

The methodologies proposed in this dissertation have provided some of the groundwork towards an *meme-centric* framework for solving complex optimization problems, by leveraging the meme co-adaptive nature of memplexes and the effective use of memplex adaptive learning and gene-meme co-evolution techniques on evaluative feedback and genetic structure examined online while the search progresses. Future research will continue to focus on the challenges posed by optimization problems in real-world scenarios with multi-criteria, multi-objective, uncertain and dynamic environment, and further extends the practicality of the memplex-based framework. A summary of several potential future works is outlined below.

6.2.1 Multi-objective/Constrained Optimization

Optimization problems in real-world applications often come in more complex scenario than single-objective optimization as presented in CVRP and VRPSD that we studied in this dissertation. For example, given the multi-objective nature of VRPSD (i.e., minimizing expected total traveled distance and at the same time maximizing the route reliability of a solution for VRPSD), the problem can be formulated as a multi-objective optimization problem. In our current approach of SAMRS, the robust solution search scheme in combination with a coevolved safety stock value in the genetic solution has the effect of producing robust solution with high route reliability at the same as it minimizes the expected total traveled distance. Nevertheless, it is important as well as challenging to place future research efforts on the topics of taking a memplex-based approach on multi-objective optimization problems and/or constraints optimization problems such as in VRPSD.

6.2.2 Optimization of Dynamic Problems

The memplex-based approaches studied so far have been used for the CVRP and VRPSD which are static by nature. They are generally not directly applicable in the context of a real-world on-line problems such as *Dynamic Vehicle Routing Problem* (DVRP), *Dynamic Travelling Salesman Problem* (DTSP). Future research work on memplex towards dynamic problem optimization is both significant and challenging. The dynamic environment often leads to very high computational cost. Moreover, the computational effort may be wasted trying to locate an optimal or near-optimal solution in a real-time setting since new customer requests may make the already found solution sub-optimal. Furthermore, application of re-optimization each time upon a new request only seems computationally tractable in problems of the low degree of dynamism. Lastly, while the current state-of-the-art algorithms for solving dynamic problems are often tied to extensive a-priori information such as the likelihood of a certain customer requiring service upon a certain day and time, such detailed information will not be available in most dynamic cases.

6.2.3 Continuous Optimization

In the present study, attention for memplex-based memetic computation has been placed solely on discrete optimization problems, in particular combinatorial optimization problems like the CVRP and VRPSD. On the other hand, memplex adaptation and evolution do apply to both continuous and discrete problems in general, complex continuous optimization in the real-world include problems involving computational intensive simulation such as computational fluid dynamics, computational chemistry, computational biology, etc. This poses as another area of notable future search for memplex-based memetic framework, where the main challenge would be on investigating effective memplex adaptation and learning mechanisms for search operations on the continuous variables.

6.2.4 Theoretical Study on Memeplex Emergence

The memeplex representation and its meme co-adaptive nature leveraged by the hybrids studied in this dissertation have been shown to be very effective in solution convergence quality, reliability and velocity. On the other hand, the emergent properties of memeplex have yet be to rigorously analyzed to enhance our understanding on memeplex emergence within these hybrids, and help to answer the following issues:

- What memes are usually activated after which meme?
- Are there any meme that is more useful at a particular stage of the evolution?
- Are there any meme that is particularly ineffective and is seldom activated?
- How to quantitatively measure the memetic network sub-patterns emerging while the search is in progress?

To fully analyze and understand these issues in general terms, and not limited to the CVRP and VRPSD problem instances considered, we believe a comprehensive study would be necessary in future research.

6.2.5 Towards Memetic Automaton

From the formalism of simple hybrids to adaptive hybrids, it is evident that algorithms seek to achieve greater level of adaptivity to address the ever increasing complexity and dynamic nature of problem-solving. The aim of memetic computation should therefore culminate into a meme-centric framework that seamlessly integrates memes into units of domain information useful for problem-solving. Taking meme as focal point of interest in the context of computational intelligence, *memetic automaton* is defined as an adaptive entity that is self-contained, uses memes as building blocks of information that facilitates problem-solving. From a software perspective, a memetic automaton is a software agent capable of autonomic behavior

during problem-solving [220, 2]. In general, the term “agent” refers to software entity that can interact with the surroundings and adapt to a complex dynamic environment. Hence, intelligent agents are able to acquire increasing level of capability through their interactions by updating its meme pool [201]. This acquired intelligence helps the agent adapts to its changing surroundings. From such perspective, memetic automaton is deemed as a framework that seamlessly espouses the notion of memetic representation and memetic evolution. It is suggestive of the fact that a memetic automaton will serve as a powerful embodiment of mechanisms associated to automated knowledge extraction and transfers [113]. This in our mind will be a strong supportive pillar of future memetic computation research.

References

- [1] Logistics in china, 2008. <http://www.kpmg.com.cn/ /LogisticsChina.pdf>.
- [2] G. Acampora, V. Loia, and M. Gaeta. Exploring e-learning knowledge through ontological memetic agents. *Computational Intelligence Magazine*, 5(2):66–77, 2010.
- [3] S. F. Adra, T. J. Dodd, I. A. Griffin, and P. J. Fleming. Convergence acceleration operator for multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 13(4), 2009.
- [4] N. M. Alexandrov, J. E. Dennis, R. M. Lewis, and V. Torczon. A trust-region framework for managing the use of approximation models in optimization. *Structural and Multidisciplinary Optimization*, 15(1):16–23, 1998.
- [5] A. S. Alfa, S. S. Heragu, and M. Chen. A 3-opt based simulated annealing algorithm for vehicle routing problem. *Computers & Operations Research*, 21:635–639, 1991.
- [6] G. B. Alvarenga, G. R. Mateus, and G. de Tomi. Finding near optimal solutions for vehicle routing problems with time windows using hybrid genetic algorithm. In *International Workshop on Freight Transportation*, page 113, 2003.
- [7] D. Applegate, R. Bixby, V. Chvatal, and W. Cook. Implementing the dantzig-fulkersonjohnson algorithm for large traveling salesman problems. *Mathematical Programming Series B*, 97(1):91C153, 2003.
- [8] P. Augerat, J. M. Belenguer, E. Benavent, A. Corber, D. Naddef, and G. Rinaldi. Computational results with a branch and cut code for the capacitated vehicle routing problem. *Research Report 949-M, Universite Joseph Fourier, Grenoble, France*, 1995.

- [9] J. M. Baldwin. *Adaptive individuals in evolving populations: models and algorithms*, chapter A new factor in evolution. Addison-Wesley Longman Publishing Co., Inc., 1996.
- [10] J. F. Bard and L. Huang. A branch and cut algorithm for the vrp with satellite facilities. *IIE Transactions*, 30(9):821–834, September 1998.
- [11] J. Berger and M. Barkaoui. A hybrid genetic algorithm for the capacitated vehicle routing problem. In *Genetic and Evolutionary Computation Conference*, volume 2723/2003, pages 646–656. Springer Berlin / Heidelberg, 2003.
- [12] S. Bernhard and K. Martin. A model for the dynamic interaction between evolution and learning. *Neural Processing Letter*, 10(3):181–193, 1999.
- [13] D. J. Bertsimas. A vehicle routing problem with stochastic demand. *Operations Research*, 40(3):574 – 585, 1992.
- [14] H. G. Beyer and H. P. Schwefel. Evolution strategies - a comprehensive introduction. *Natural Computing*, 1(1), 2002.
- [15] L. Bianchi, M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto. Metaheuristics for the vehicle routing problem with stochastic demands. *Parallel problem solving from nature: PPSN VIII*, 3242:1 – 10, 2004.
- [16] S. J. Blackmore. *The Meme Machine*. New York: Oxford University Press, 1999.
- [17] David. Blitz. *Emergent evolution : qualitative novelty and the levels of reality / by David Blitz*. Kluwer Academic Publishers, Dordrecht ; Boston :, 1992.
- [18] Christian Blum, Carlos Cotta, Antonio J. Fernndez, Jos E. Gallardo, and Monaldo Mastrolilli. Hybridizations of metaheuristics with branch & bound derivates. *Hybrid Metaheuristics*, page 85C116, 2008.
- [19] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *Swarm intelligence: from natural to artificial systems*. Oxford University Press, Inc., New York, NY, USA, 1999.

- [20] E. Borenstein, I. Meilijson, and E. Ruppin. The effect of phenotypic plasticity on evolution in multipeaked fitness landscapes. *Journal of Evolutionary Biology*, 19(5):1555–1570, 2006.
- [21] P. A. N. Bosman and E. D. De Jong. Combining gradient techniques for numerical multi-objective evolutionary optimization. In *Genetic and Evolutionary Computation Conference*, pages 627–634, 2006.
- [22] E. Burke, S. Gustafson, G. Kendall, and N. Krasnogor. *Parallel problem solving from nature: PPSN VII*, chapter Advanced Population Diversity Measures in Genetic Programming, pages 341–350. 2002.
- [23] A. Caponio, G. L. Cascella, F. Neri, N. Salvatore, and M. Sumner. A fast adaptive memetic algorithm for online and offline control design of pmsm drives. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(1):28–41, 2007.
- [24] A. Caponio and F. Neri. Integrating cross-dominance adaptation in multi-objective memetic algorithms. *Studies in Computational Intelligence*, 171:325–351, 2009.
- [25] A. Caponio, F. Neri, and V. Tirronen. Super-fit control adaptation in memetic differential evolution frameworks. *Soft Computing*, 13(8):811–831, 2009.
- [26] S. H. Chen, P. C. Chang, Q. Zhang, and C. B. Wang. A guided memetic algorithm with probabilistic models. *International Journal of Innovative Computing, Information and Control*, 5(12):4753–4764, 2009.
- [27] X. Chen, Y. S. Ong, and M. H. Lim. Cooperating memes for vehicle routing problems. *International Journal of Innovative Computing, Information and Control*, 7(11):1–10, 2011.
- [28] Xianshun Chen, Yew-Soon Ong, Meng-Hiot Lim, and Kay Chen Tan. A multi-facet survey on memetic computation. *IEEE Transactions on Evolutionary Computation*, 15(5):591–607, 2011.
- [29] S. S. Choi and B. R. Moon. A graph-based lamarckian-baldwinian hybrid for the sorting network problem. *IEEE Transactions on Evolutionary Computation*, 9:105–114, 2005.

- [30] C. Christiansen and J. Lysgaard. A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. *Operational Research Letters*, 35(6):773–781, 2007.
- [31] N. Christofides and S. Eilon. An algorithm for the vehicle dispatching problem. *Operational Research Quarterly*, 20:309–318, 1969.
- [32] N. Christofides, A. Mingozzi, and P. Toth. The vehicle routing problem. *Combinatorial Optimization*, pages 315–338, 1979.
- [33] C. A. Coello Coello, G. Pulido, and E. Montes. *Current and Future Research Trends in Evolutionary Multiobjective Optimization*, pages 213–231. Advanced Information and Knowledge Processing. Springer London, 2005.
- [34] W. Conley. Simulation optimization and correlation with multi stage monte carlo optimization. *International Journal of Systems Science*, 38(12):1013–1019, 2007.
- [35] J. F. Cordeau, M. Gendreau, and G. Laporte. A tabu search heuristic for the periodic and multi-depot vehicle routing problems. *Networks*, 30:105–119, 1994.
- [36] J.F. Cordeau, M. Gendreau, A. Hertz, G. Laporte, and J.S. Sormany. New heuristics for the vehicle routing problem. *Logistic Systems: Design and Optimization*, page 279C298, 2005.
- [37] V. Cutello, N. Krasnogor, G. Nicosia, and M. Pavone. *Adaptive and Natural Computing Algorithms*, chapter Immune Algorithm Versus Differential Evolution: A Comparative Case Study Using High Dimensional Function Optimization, pages 93–101. 2007.
- [38] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science*, 6(1):pp. 80–91, 1959.
- [39] D. Dasgupta. Advances in artificial immune systems. *Computational Intelligence Magazine*, 1(4):40–49, 2006.
- [40] R. Dawkins. *The Selfish Gene*. Clarendon Press, Oxford, 1989.

- [41] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm : Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
- [42] B. D. Diaz. Vrp benchmarks. World Wide Web electronic publication, 2009.
- [43] M. Dorigo and T. Sttze. *Ant Colony Optimization*. MIT Press, 2004.
- [44] S. Droste, T. Jansen, and I. Wegener. On the analysis of the (1+ 1) evolutionary algorithm. *Theoretical Computer Science*, 276(1):51–81, 2002.
- [45] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby. Self-adaptive baldwinian search in hybrid genetic algorithms. *Computational Intelligence, Theory and Applications*, pages 597–602, 2006.
- [46] E. Elbeltagi, T. Hegazy, and D. Grierson. Comparison among five evolutionary-based optimization algorithms. *Advanced Engineering Informatics*, 19(1):43 – 53, 2005.
- [47] S. Elfwing, E. Uchibe, K. Doya, and H. I. Christensen. Evolutionary development of hierarchical learning structures. *IEEE Transactions on Evolutionary Computation*, 11(2):249–264, 2007.
- [48] A. M. Farahmand, M. N. Ahmadabadi, C. Lucas, and B. N. Araabi. Interaction of culture-based learning and cooperative co-evolution and its application to automatic behavior-based system design. *IEEE Transactions on Evolutionary Computation*, 14(1):23–57, 2010.
- [49] M. L. Fisher. Optimal Solution of Vehicle Routing Problems Using Minimum K-Trees. *Operations Research*, 42(4):626–642, July 1994.
- [50] G. Fleury, P. Lacomme, and C. Prins. Evolutionary algorithms for stochastic arc routing problems. *EvoWorkshops on EvoCoMnet, EvoFIN, EvoIASP, EvoINTERACTION, EvoMUSART, EvoSTOC and EvoTransLog*, 3005:501 – 512, 2004.
- [51] C. A. Floudas. *Deterministic Global Optimization: Theory, Methods and Application*, volume 37 of *Nonconvex Optimization and Its Applications*. Springer-Verlag New York, Inc., 2005.

- [52] L. J. Fogel. *Intelligence through simulated evolution: forty years of evolutionary programming*. Wiley Series On Intelligent Systems. John Wiley & Sons, Inc., 1999.
- [53] J. Gallardo, C. Cotta, and A. J. Ferndez. Finding low autocorrelation binary sequences with memetic algorithms. *Applied Soft Computing*, 9(4):1252 – 1262, 2009.
- [54] Cristian Andrs Gallo, Jessica Andrea Carballido, and Ignacio Ponzoni. Bihea: A hybrid evolutionary approach for microarray biclustering. In *Advances in Bioinformatics and Computational Biology*, volume 5676 of *Lecture Notes in Computer Science*, pages 36–47. Springer, 2009.
- [55] L. M. Gambardella, E. Taillard, and G. Agazzi. Macs-vrptw: A multiple ant colony system for vehicle routing problems with time windows. *New Ideas in Optimization*, pages 63–76, 1999.
- [56] S. García, J. C. Cano, and F. Herrera. A memetic algorithm for evolutionary prototype selection: A scaling up approach. *Pattern Recognition*, 41(8):2693–2709, 2008.
- [57] M. R. Garey and D. S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman, January 1979.
- [58] M. Gendreau, A. Hertz, and G. Laporte. A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10):1276–1290, 1994.
- [59] M. Gendreau, G. Laporte, and J. Y. Potvin. *Local search in combinatorial optimization*. Princeton : Princeton University Press, c2003, 2003. Reprint. Originally published: New York : Wiley, 1997.
- [60] B. E. Gillett and L. R. Miller. A heuristic algorithm for the vehicle-dispatch problem. *Operations Research*, 22(2):340–349, 1974.
- [61] C. K. Goh, Y. S. Ong, and K. C. Tan. *Multi-Objective Memetic Algorithms*. Springer Publishing Company, Incorporated, 2009.

- [62] C. K. Goh and K. C. Tan. *Evolutionary Computation in Dynamic and Uncertain Environments*, volume 51, chapter Evolving the Tradeoffs between Pareto-Optimality and Robustness in Multi-Objective Evolutionary Algorithms, pages 457–478. Springer, 2007.
- [63] B. L. Golden, E. A. Wasil, J. P. Kelly, and I. M. Chao. The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. *Fleet management and logistics*, pages 33–56, 1998.
- [64] Jeffrey Goldstein. Emergence as a Construct: History and Issues. *Emergence*, 1(1):49–72, 1999.
- [65] S. B. Gottfried and J. Weisman. *Introduction to Optimization Theory*. Prentice-Hall international series in industrial and systems engineering. Englewood Cliffs, N.J., Prentice-Hall, 1973.
- [66] A. Guillen, H. Pomares, J. Gonzalez, I. Rojas, L. J. Herrera, and A. Prieto. Parallel multi-objective memetic rbfnn design and feature selection for function approximation problems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4507:341–350, 2007.
- [67] G. Gutin and D. Karapetyan. Generalized traveling salesman problem reduction algorithms. *Algorithmic Operations Research*, 4:144–154, 2008.
- [68] S.D. Handoko, C.K. Kowh, and Y. S. Ong. Using classification for constrained memetic algorithm: A new paradigm. In *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, pages 547–552, 2008.
- [69] S.D. Handoko, C.K. Kwoh, and Y. S. Ong. Feasibility structure modeling: An effective chaperon for constrained memetic algorithms. *IEEE Transactions on Evolutionary Computation*, 14(5):740 – 758, 2009.
- [70] W. E. Hart. *Adaptive global optimization with local search*. PhD thesis, University of California at San Diego, 1994.

- [71] R. Hassin and S. Rubinstein. On the complexity of the k-customer vehicle routing problem. *Operations Research Letters*, 33(1):71 – 76, 2005.
- [72] F. Heylighen and K. Chielens. *Encyclopedia of Complexity and System Science*, chapter Cultural Evolution and Memetics. Springer, 2008.
- [73] H. Ishibuchi, Y. Hitotsuyanagi, and Y. Nojima. An empirical study on the specification of the local search application probability in multiobjective memetic algorithms. In *IEEE Congress on Evolutionary Computation*, pages 2788–2795, 2008.
- [74] H. Ishibuchi, S. Kaige, and K. Narukawa. *Evolutionary Multi-Criterion Optimization*, volume 3410 of *Lecture Notes in Computer Science*, chapter Comparison Between Lamarckian and Baldwinian Repair on Multiobjective 0/1 Knapsack Problems, pages 370–385. Springer, 2005.
- [75] H. Ishibuchi and T. Murata. A multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 28(3):392–403, 1998.
- [76] H. Ishibuchi, N. Tsukamoto, and Y. Nojima. Diversity improvement by non-geometric binary crossover in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 14(6):985 – 998, 2010.
- [77] H. Ishibuchi and T. Yamamoto. Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining, fuzzy sets and systems. *Genetic Fuzzy Systems: New Developments*, 141(1):59–88, 2004.
- [78] H. Ishibuchi, T. Yoshida, and T. Murata. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *IEEE Transactions on Evolutionary Computation*, 7:204–223, 2002.
- [79] W. Jakob. Towards an adaptive multimeme algorithm for parameter optimisation suiting the engineers’ needs. In *Parallel Problem Solving From Nature*, pages 132–141, 2006.
- [80] W. Jakob. A cost-benefit-based adaptation scheme for multimeme algorithms. In *Parallel Processing and Applied Mathematics*, pages 509–519, 2007.

- [81] P. Ji, Y. Wu, H. Liu, and H. Wu. The vehicle routing problem with time-varying travel times and a solution method. *International Journal of Innovative Computing, Information and Control*, 5:1001–1012, 2009.
- [82] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 9(1):3–12, 2005.
- [83] Y. Jin. *Multi-Objective Machine Learning*. Studies in Computational Intelligence. Springer, 2006.
- [84] Y. Jin and J. Branke. Evolutionary optimization in uncertain environments-a survey. *IEEE Transactions on Evolutionary Computation*, 9(3):303–317, 2005.
- [85] A. Juan, J. Faulin, S. Grasman, D. Riera, J. Marull, and C. Mendez. Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands. *Transportation Research Part C: Emerging Technologies*, In Press, Corrected Proof, 2010.
- [86] Y. Kameya, J. Kumagai, and Y. Kurata. Accelerating genetic programming by frequent subtree mining. In *Genetic and Evolutionary Computation Conference*, pages 1203–1210. ACM, 2008.
- [87] R. G. Kang and C. Y. Jung. Neural information processing. 4234:789–796, 2006.
- [88] R. G. Kang and C. Y. Jung. *PRICAI 2006: Trends in Artificial Intelligence*, volume 4099, chapter The Optimal Solution of TSP Using the New Mixture Initialization and Sequential Transformation Method in Genetic Algorithm, pages 1181–1185. Springer, 2006.
- [89] G. Kendall and E. Soubeiga. Choice function and random hyperheuristics. In *Proceedings of the fourth Asia-Pacific Conference on Simulated Evolution And Learning, SEAL*, pages 667–671. Springer, 2002.
- [90] S. Kersting, G. Raidl, and I. Ljubic. *Applications of Evolutionary Computing*, chapter A Memetic Algorithm for Vertex-Biconnectivity Augmentation, pages 102–111. Lecture Notes in Computer Science. Springer, 2002.

- [91] G. W. Klau, I. Ljubic, A. Moser, P. Mutzel, P. Neuner, U. Pferschy, G. Raidl, and R. Weiskircher. Combining a memetic algorithm with integer programming to solve the prize-collecting steiner tree problem. In *Genetic and Evolutionary Computation Conference*, pages 1304–1315, 2004.
- [92] J Knowles and D. Corne. M-paes: a memetic algorithm for multiobjective optimization. volume 1, pages 325–332, 2000.
- [93] J. Knowles, D. Corne, and K. Deb. *Multiobjective Problem Solving from Nature: From Concepts to Applications*. Natural Computing Series. Springer, 2008.
- [94] O. Kramer and H. Danielsiek. Dbscan-based multi-objective niching to approximate equivalent pareto-subsets. In *Genetic and Evolutionary Computation Conference*, pages 503–510, 2010.
- [95] N. Krasnogor. Self-generating metaheuristics in bioinformatics: The protein structure comparison case. In *Genetic Programming and Evolvable Machines*, pages 181–201. Kluwer academic Publishers, 2004.
- [96] N. Krasnogor. Self generating metaheuristics in bioinformatics: The proteins structure comparison case. *Genetic Programming and Evolvable Machines*, 5(2):181–201, 2004.
- [97] N. Krasnogor, B. P. Blackburne, E. K. Burke, and J. D. Hirst. Multimeme algorithms for protein structure prediction. In *Proceedings of the Parallel Problem Solving from Nature VII. Lecture Notes in Computer Science*, pages 769–778. Springer-Verlag, 2002.
- [98] N. Krasnogor and S. Gustafson. A study on the use of self-generation in memetic algorithms. *Natural Computing*, 3(1):53C76, 2004.
- [99] N. Krasnogor and J. Smith. A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *IEEE Transactions on Evolutionary Computation*, 9(5):474–488, 2005.
- [100] N. Krasnogor and J. Smith. Memetic algorithms: The polynomial local search complexity theory perspective. *Journal of Mathematical Modelling and Algorithms*, 7(1):3–24, 2008.

- [101] M. Kubiak. Systematic construction of recombination operators for the vehicle routing problem. *Foundations of Computing and Decision Sciences*, 29:205–226, 2004.
- [102] M. Kubiak. Distance measures and fitness-distance analysis for the capacitated vehicle routing problem. In *Metaheuristics*, volume 39, pages 345–364. Springer US, 2007.
- [103] G. Laporte. Fifty years of vehicle routing. *Transportation Science*, 43:408–416, November 2009.
- [104] G. Laporte, F. Louveaux, and L. Van Hamme. An integer l-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50(3):415 – 423, 2002.
- [105] G. Laporte, F. Louveaux, and H. Mercure. Models and exact solutions for a class of stochastic location-routing problems. *European Journal of Operational Research*, 39(1):71 – 78, 1989.
- [106] A. Lara, G. Sanchez, C. A. C. Coello, and O. Schtze. Hcs: A new local search strategy for memetic multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 14(1):112–132, 2010.
- [107] P. Larraaga. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2002.
- [108] M. N. Le and Y. S. Ong. A frequent pattern mining algorithm for understanding genetic algorithms. In *International Conference on Intelligent Computing: Advanced Intelligent Computing Theories and Applications*, pages 131–139, 2008.
- [109] M. N. Le, Y. S. Ong, Y. Jin, and B. Sendhoff. Lamarckian memetic algorithms: local optimum and connectivity structure analysis. *Memetic Computing*, 1(3):175–190, 2009.
- [110] M. N. Le, Y. S. Ong, and Q. H. Nguyen. Optinformatics for schema analysis of binary genetic algorithms. In *Genetic and Evolutionary Computation Conference*, pages 1121–1122, 2008.
- [111] G. H. Lewes. *Problems of Life and Mind*, volume 2. Kegan Paul, Trench, Turbner, & Co., London, 1875.

- [112] Y. Li, X. Guan, and X. Luo. Robust exponential stability criteria of uncertain stochastic systems with time-varying delays. *International Journal of Systems Science*, 42(4):601–608, 2011.
- [113] F. Liang, Y. S. Ong, A.H. Tan, and X.S. Chen. Towards human-like social multi-agents with memetic automaton. In *IEEE Congress on Evolutionary Computation*, pages 1092 – 1099, 2011.
- [114] J. Liang and J. Lam. Robust state estimation for stochastic genetic regulatory networks. *International Journal of Systems Science*, 41(1):47–63, 2010.
- [115] D. Lim, Y. Jin, Y. S. Ong, and B. Sendhoff. Generalizing surrogate-assisted evolutionary computation. *IEEE Transactions on Evolutionary Computation*, 14(3):329–355, 2010.
- [116] M.H. Lim, S. Gustafson, N. Krasnogor, and Y.S. Ong. Editorial to the first issue. *Memetic Computing*, 1(1):1–2, 2009.
- [117] D Liu, K. C. Tan, C. K. Goh, and W. K. Ho. A multiobjective memetic algorithm based on particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(1):42–50, 2007.
- [118] G. Mayley. The evolutionary cost of learning. In *the Fourth International Conference on Simulation of Adaptive behavior*, pages 458–467. MIT Press, 1996.
- [119] J. E. Mendoza, B. Castanier, C. Guéret, A. L. Medaglia, and N. Velasco. A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, 37(11):1886 – 1898, 2010.
- [120] P. Merz. *Recent Advances in Memetic Algorithms*, chapter NK-Fitness Landscapes and Memetic Algorithms with Greedy Operators and k-opt Local Search, pages 209–228. 2005.
- [121] D. Mester and O. Bräysy. Active guided evolution strategies for large scale capacitated vehicle routing problems. *Computers & Operations Research*, 34:2964C2975, 2007.

- [122] R. Meuth, M. H. Lim, Y. S. Ong, and D. C. Wunsch II. A proposition on memes and meta-memes in computing for higher-order learning. *Memetic Computing*, 1(2):85–100, 2009.
- [123] M. Mezmoz, N. Melab, and E. G. Talbi. Combining metaheuristics and exact methods for solving exactly multi-objective problems on the grid. *Journal of Mathematical Modelling and Algorithms*, 6(3):393–409, 2007.
- [124] K. Miettinen. *Evolutionary Algorithms in Engineering and Computer Science: Recent Advances in Genetic Algorithms, Evolution Strategies, Evolutionary Programming*, GE. John Wiley & Sons, Inc., 1999.
- [125] R. Mills and R. A. Watson. On crossing fitness valleys with the baldwin effect. In *International Conference on the Simulation and Synthesis of Living Systems*, pages 493–499, 2006.
- [126] M. Mitchell. *An Introduction to Genetic Algorithms (Complex Adaptive Systems)*. The MIT Press, 1998.
- [127] D. Molina. Adaptive local search parameters for real-coded memetic algorithm. In *IEEE Congress on Evolutionary Computation*, pages 888–895, 2005.
- [128] D. Molina, M. Lozano, and F. Herrera. Memetic algorithm with local search chaining for continuous optimization problems: A scalability test. In *International Conference on Intelligent Systems Design and Applications*, pages 1068–1073, 2009.
- [129] D. Molina, M. Lozano, and F. Herrera. Ma-sw-chains: Memetic algorithm based on local search chains for large scale continuous global optimization. In *IEEE Congress on Evolutionary Computation*, pages 1–8, 2010.
- [130] James Montgomery, Carole Fayad, and Sanja Petrovic. *Ant Colony Optimization and Swarm Intelligence*, chapter Solution Representation for Job Shop Scheduling Problems in Ant Colony Optimisation. Lecture Notes in Computer Science. Springer, 2006.
- [131] K. Morikawa, S. Ozawa, and S. Abe. Tuning membership functions of kernel fuzzy classifiers by maximizing margins. *Memetic Computing*, 1(3):221–228, 2009.

- [132] R. W. Morrison. Dispersion-based population initialization. In *Genetic and Evolutionary Computation Conference*, volume 2723, pages 1210–1221. Springer-Verlag, 2003.
- [133] P. Moscato. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Technical Report Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA, 1989.
- [134] F. Neri, Niko Kotilainen, and Mikko Vapa. A memetic-neural approach to discover resources in p2p networks. In *Recent Advances in Evolutionary Computation for Combinatorial Optimization*, pages 113–129. 2008.
- [135] F. Neri, V. Tirronen, T. Karkkainen, and T. Rossi. Fitness diversity based adaptation in multimeme algorithms: A comparative study. In *IEEE Congress on Evolutionary Computation*, pages 2374–2381, 2008.
- [136] Q. H. Nguyen, Y. S. Ong, and M. H. Lim. Non-genetic transmission of memes by diffusion. In *Genetic and Evolutionary Computation Conference*, pages 1017–1024, New York, NY, USA, 2008. ACM.
- [137] Q. H. Nguyen, Y. S. Ong, and M. H. Lim. A probabilistic memetic framework. *IEEE Transactions on Evolutionary Computation*, 13(3):604–623, 2009.
- [138] Q. H. Nguyen, Y. S. Ong, M. H. Lim, and N. Krasnogor. Adaptive cellular memetic algorithms. *Evolutionary Computation*, 17(2):231–256, 2009.
- [139] Q.H. Nguyen, Y.S. Ong, and N. Krasnogor. A study on the design issues of memetic algorithm. In *IEEE Congress on Evolutionary Computation*, pages 2390–2397, 2008.
- [140] Q.H. Nguyen, Y.S. Ong, and J.L. Kuo. A hierarchical approach to study the thermal behavior of protonated water clusters $h+(h_2o)_n$. *Journal Chemical Theory & Computation*, 5(10):2629–2639, 2009.
- [141] R. Niedermeier. *Invitation to Fixed Parameter Algorithms*. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2006.
- [142] N. Noman and H. Iba. Accelerating differential evolution using an adaptive local search. *IEEE Transaction on Evolutionary Computation*, 12(1):107C125, 2008.

- [143] C. Novoa, R. Berger, J. Linderoth, and R. Storer. A set-partitioning-based model for the stochastic vehicle routing problem. *Technical Report. Industrial and Systems Engineering, Lehigh University*, 2006.
- [144] C. Novoa and R. Storer. An approximate dynamic programming approach for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 196(2):509 – 515, 2009.
- [145] Ana C. Olivera, Mariano Frutos, Jessica A. Carballido, Ignacio Ponzoni, and Nélide B. Brignole. Bus network scheduling problem: Grasp + eas with pisa * simulation. In *IWANN '09: Proceedings of the 10th International Work-Conference on Artificial Neural Networks*, pages 1272–1279, Berlin, Heidelberg, 2009. Springer-Verlag.
- [146] Intergovernmental Panel on Climate Change. *IPCC Fourth Assessment Report: Mitigation of Climate Change*, chapter Transport and its Infrastructure. 2007. Retrieved 2009-05-26.
- [147] Y. S. Ong and A. J. Keane. Meta-lamarckian learning in memetic algorithms. *IEEE Transactions on Evolutionary Computation*, 8(2):99–110, April 2004.
- [148] Y. S. Ong, M. H. Lim, and X. Chen. Memetic computation – past, present and future. *Computational Intelligence Magazine*, 5(2):24–36, 2010.
- [149] Y. S. Ong, M. H. Lim, N. Zhu, and K. W. Wong. Classification of adaptive memetic algorithms: a comparative study. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 36(1):141–152, Feb. 2006.
- [150] Y. S. Ong, P. B. Nair, and A. J. Keane. Evolutionary optimization of computationally expensive problems via surrogate modeling. *American Institute of Aeronautics and Astronautics Journal*, 2003.
- [151] Y. S. Ong, P. B. Nair, A. J. Keane, and K. W. Wong. Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems. In *Knowledge Incorporation in Evolutionary Computation*, editor: Y. Jin, *Studies in Fuzziness and Soft Computing Series*, pages 307–332. Springer Verlag, 2004.

- [152] Y. S. Ong, P. B. Nair, and K. Y. Lum. Max-min surrogate-assisted evolutionary algorithm for robust design. *IEEE Transactions on Evolutionary Computation*, 10(4):392–404, Aug. 2006.
- [153] Y.S. Ong and A.J. Keane. A domain knowledge based search advisor for design problem solving environments. *Engineering Applications of Artificial Intelligence*, 15(1):105–116, 2002.
- [154] Y.S. Ong, N. Krasnogor, and H. Ishibuchi. Special issue on memetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(1):2–5, 2007.
- [155] Y.S. Ong, M.H. Lim, F. Neri, and H. Ishibuchi. Special issue on 'emerging trends in soft computing - memetic algorithm'. *Soft Computing*, 13(8), 2009.
- [156] I. H. Osman. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of Operations Research*, 41(4):421–451, December 1993.
- [157] E. Özcan, B. Bilgin, and E. E. Korkmaz. A comprehensive analysis of hyper-heuristics. *Intelligent Data Analysis*, 12(1):3–23, 2008.
- [158] I. Paenke, J. Branke, and Y. Jin. On the influence of phenotype plasticity on genotype diversity. In *Foundations of Computational Intelligence*, pages 33–40, 2007.
- [159] I. Paenke, T. J. Kawecki, and B. Sendhoff. The influence of learning on evolution: A mathematical framework. *Artificial Life*, 15(2):227–245, 2009.
- [160] I. Paenke, B. Sendhoff, J. Rowe, and C. Fernando. *Advances in Artificial Life*, chapter On the Adaptive Disadvantage of Lamarckianism in Rapidly Changing Environments, pages 355–364. 2007.
- [161] T. Park and K. R. Ryu. A dual-population genetic algorithm for adaptive diversity control. *IEEE Transactions on Evolutionary Computation*, 14(6):865–884, 2010.
- [162] F. B. Pereira, J. Tavares, P. Machado, and E. Costa. Gvr: a new genetic representation for the vehicle routing problem. In *Irish Conference on Artificial Intelligence and Cognitive Science*, pages 95–102. Springer-Verlag, 2002.

- [163] Y. Petalas, K. Parsopoulos, and M. Vrahatis. Memetic particle swarm optimization. *Annals of Operations Research*, 156(1):99–127, 2007.
- [164] D. Pisinger and S. Röpkke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34:2403C2435, 2007.
- [165] R. Poli, J. Kennedy, and T. Blackwell. Particle swarm optimization. *Swarm Intelligence*, 1(1):33–57, June 2007.
- [166] R. Poli, W. Langdon, and N. McPhee. *A field guide to genetic programming*. Lulu.com, 2008.
- [167] Jean-Yves Potvin. *A Review of Bio-inspired Algorithms for Vehicle Routing*, pages 1–34. 2009.
- [168] Jean-Yves Potvin. State-of-the art review - evolutionary algorithms for vehicle routing. *INFORMS Journal on Computing*, 21:518–548, 2009.
- [169] K. Price, R. Storn, and J. Lampinen. *Differential Evolution A Practical Approach to Global Optimization*. Natural Computing Series. Springer-Verlag, 2005.
- [170] C. Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002, October 2004.
- [171] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama. A novel population initialization method for accelerating evolutionary algorithms. *Computers & Mathematics with Applications*, 53(10):1605–1614, 2007.
- [172] T. Ralphs. Classical vrp benchmarks, 2010.
- [173] A. Ratle. Kriging as a surrogate fitness landscape in evolutionary optimization. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 15(1):37–49, 2001.
- [174] C. Rego and C. Rouchairol. *Metaheuristics: Theory and Applications*, chapter A Parallel Tabu Search Algorithm Using Ejection Chains for the Vehicle Routing Problem, pages 661–675. Kluwer, Boston, 1996.

- [175] M. Reimann, K. Doerner, and R.F. Hartl. D-ants: savings based ants divide and conquer the vehicle routing problem. *Computers & Operations Research*, 31:563C591, 2004.
- [176] P. P. Repoussis, C. D. Tarantilis, and G. Ioannou. Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. *IEEE Transactions on Evolutionary Computation*, 13(3):624–647, 2009.
- [177] Panagiotis P. Repoussis, Christos D. Tarantilis, and George Ioannou. *Bio-inspired Algorithms for the Vehicle Routing Problem*, chapter An Evolutionary Algorithm for the Open Vehicle Routing Problem with Time Windows, pages 55–75. 2009.
- [178] Jean-Michel Richer, Adrien Goëffon, and Jin-Kao Hao. A memetic algorithm for phylogenetic reconstruction with maximum parsimony. In *European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, pages 164–175, Berlin, Heidelberg, 2009. Springer-Verlag.
- [179] Valentin Robu, Harry Halpin, and Hana Shepherd. Emergence of consensus and shared vocabularies in collaborative tagging systems. *ACM Transactions on the Web (TWEB)*, 3(4):1–34, September 2009.
- [180] Y. Rochat and E. D. Taillard. Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics I*, pages 147–167, 1995.
- [181] Choi S. S., Jung K., and Moon B. R. Lower and upper bounds for linkage discovery. *IEEE Transactions on Evolutionary Computation*, 13(2):201–216, 2009.
- [182] Choi S. S., Kwon Y. K., and Moon B. R. Properties of symmetric fitness functions. *IEEE Transactions on Evolutionary Computation*, 11(6):743–757, 2007.
- [183] T. Sasaki and M. Tokoro. Adaptation toward changing environments: why darwinian in nature? In *European Conference on Artificial Life*, pages 145–153. Brighton, UK:MIT press, 1997.
- [184] T. Sasaki and M. Tokoro. Comparison between lamarckian and darwinian evolution on a model using neural networks and genetic algorithms. *Knowledge and Information Systems*, 2(2):201–222, 2000.

- [185] M. D. Schmidt and H. Lipson. Coevolution of fitness predictors. *IEEE Transactions on Evolutionary Computation*, 12(6):736–749, 2008.
- [186] J. Schonberger. *Operational Freight Carrier Planning*, chapter Memetic Algorithm Vehicle Routing, pages 65–76. Springer Berlin Heidelberg, 2006.
- [187] N. Secomandi. Comparing neuro dynamic programming algorithm for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 27:1201 – 1225, 2000.
- [188] N. Secomandi and F. Margot. Reoptimization approaches for the vehicle-routing problem with stochastic demands. *Operational Research*, pages 1 – 17, 2008.
- [189] P. Shaw. Using constraint programming and local search methods to solve vehicle routing problems. *International Conference on Principles and Practice of Constraint Programming*, pages 417–431, 1998.
- [190] Z. Shen, F. Ordóñez, and M. M. Dessouky. The Stochastic Vehicle Routing Problem for Minimum Unmet Demand. *Optimization and Logistics Challenges in the Enterprise*, pages 349–371, 2009.
- [191] A. Sinha, Y. P. Chen, and D. E. Goldberg. *Recent Advances in Memetic Algorithms*, chapter Designing Efficient Genetic and Evolutionary Algorithm Hybrids, pages 259–288. 2005.
- [192] H. Situngkir. On selfish memes: culture as complex adaptive system, 2004.
- [193] Z. Skolicki and K. De Jong. The influence of migration sizes and intervals on island models. In *Genetic and Evolutionary Computation Conference*, pages 1295–1302. ACM, 2005.
- [194] J. Smith. Co-evolving memetic algorithms: Initial investigations. In *Parallel problem solving from nature: PPSN VII*, pages 537–548, London, UK, UK, 2002. Springer-Verlag.
- [195] K. Sorensen and M. Sevaux. Ma—pm: memetic algorithms with population management. *Computers and Operations Research*, 33:1214–1225, 2006.

- [196] B. Soylu and M. Koksalan. A favorable weight-based evolutionary algorithm for multiple criteria problems. *IEEE Transactions on Evolutionary Computation*, 14(2):191–205, 2010.
- [197] J. C. Spall. *Introduction to Stochastic Search and Optimization*. John Wiley & Sons, Inc., 2003.
- [198] Dirk Sudholt. *Algorithms and Computation*, volume 4288 of *Lecture Notes in Computer Science*, chapter Local Search in Evolutionary Algorithms: The Impact of the Local Search Frequency, pages 359–368. 2006.
- [199] I. Sungur, F. Ordóñez, and MM. Dessouky. A robust optimizaiton approach for the capacitated vehicle routing problem with demand uncertainty. *IIE Transactions*, 40(5):509 – 523, 2008.
- [200] E. D. Taillard. Parallel iterative search methods for vehicle routing problems. *Networks*, 23(8):661–673, 1993.
- [201] A. H. Tan, L. Ning, and X. Dan. Integrating temporal difference methods and self-organizing neural networks for reinforcement learning with delayed evaluative feedback. *IEEE Transactions on Neural Networks*, 19(2):230–244, 2008.
- [202] K. C. Tan, C. Y. Cheong, and C. K. Goh. Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. *European Journal of Operational Research*, 177(2):813 – 839, 2007.
- [203] K. C. Tan, E. F. Khor, and T. H. Lee. *Multiobjective Evolutionary Algorithms and Applications (Advanced Information and Knowledge Processing)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.
- [204] K. C. Tan, T. H. Lee, and E. F. Khor. Evolutionary algorithms with dynamic population size and local exploration for multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 5(6):565–588, 2001.
- [205] J. Tang, M. H. Lim, and Y. S. Ong. Adaptation for parallel memetic algorithm based on population entropy. In *Genetic and Evolutionary Computation Conference*, pages 575–582, New York, NY, USA, 2006. ACM.

- [206] J. Tang, M. H. Lim, and Y. S. Ong. Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems. *Soft Computing*, 11(9):873–888, 2007.
- [207] K. Tang, Y. Mei, and X. Yao. Memetic algorithm with extended neighborhood search for capacitated arc routing problems. *IEEE Transactions on Evolutionary Computation*, 13(5):1151–1166, 2009.
- [208] C.D. Tarantilis. Solving the vehicle routing problem with adaptive memory programming methodology. *Computers & Operations Research*, 32:2309C2327, 2005.
- [209] C.D. Tarantilis and C.T. Kiranoudis. Bone route: an adaptive memory-based method for effective fleet management. *Annals of Operations Research*, 115), year = 2002:227C241.
- [210] Y. Tenne and S. W. Armfield. A memetic algorithm using a trust-region derivative-free optimization with quadratic modelling for optimization of expensive and noisy black-box functions. *Studies in Computational Intelligence*, 51:389–415, 2007.
- [211] Y. Tenne and S.W. Armfield. A versatile surrogate-assisted memetic algorithm for optimization of computationally expensive functions and its engineering applications. *Studies in Computational Intelligence*, 92:43–72, 2008.
- [212] Yoel Tenne and S. W. Armfield. A framework for memetic optimization using variable global and local surrogate models. *Soft Computing*, 13(8-9):781–793, 2009.
- [213] D. Teodorovic and G. Pavkovic. A simulated annealing technique approach to the vehicle routing problem in the case of stochastic demand. *Transportation Planning and Technology*, 16:261 – 273, 1992.
- [214] V. Tirronen, F. Neri, Tommi Karkkainen, Kirsi Majava, and Tuomo Rossi. A memetic differential evolution in filter design for defect detection in paper production. In *EvoWorkshops on EvoCoMnet, EvoFIN, EvoIASP, EvoINTERACTION, EvoMUSART, EvoSTOC and EvoTransLog*, pages 320–329, Berlin, Heidelberg, 2007. Springer-Verlag.

- [215] P. Toth and D. Vigo. *Fleet Management and Logistic*, chapter Exact solution of the vehicle routing problem, pages 1–31. Kluwer Academic Publisher, 1998.
- [216] P. Toth and D. Vigo. The granular tabu search and its application to the vrp. *INFORMS Journal on Computing*, 13(4):333–346, 1998.
- [217] S. Tsutsui and A. Ghosh. Genetic algorithms with a robust solution searching scheme. *IEEE Transactions on Evolutionary Computation*, 1(3):201 –208, September 1997.
- [218] A.S.S.M.B. Ullah, R. Sarker, D. Cornforth, and C. Lokan. An agent-based memetic algorithm (ama) for solving constrained optimization problems. In *IEEE Congress on Evolutionary Computation*, pages 999–1006. IEEE, 2008.
- [219] S. Verel, G. Ochoa, and M. Tomassini. Local optima networks of nk landscapes with neutrality. *IEEE Transactions on Evolutionary Computation (to appear)*, 2010.
- [220] D. Vernon, G. Metta, and G. Sandini. A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents. *IEEE Transactions on Evolutionary Computation*, 11(2):151–180, 2007.
- [221] H. Wang, D. Wang, and S. Yang. A memetic algorithm with adaptive hill climbing strategy for dynamic optimization problems. *Soft Computing*, 13(8-9):763–780, 2009.
- [222] X. Wang, C. Xu, and D. Yang. Disruption management for vehicle routing problem with the request changes of customers. *International Journal of Innovative Computing, Information and Control*, 5:2427–2438, 2009.
- [223] X. Wang, K. Zhang, and D. Yang. Disruption management of urgency vehicle routing problem with fuzzy time window. *ICIC Express Letters*, 3:883–890, 2009.
- [224] Y. W. Wang, C. H. Liu, and C. Y. Fan. Develop a sub-population memetic algorithm for multi-objective scheduling problems. In *the 2nd International Conference on Computer and Automation Engineering (ICCAE)*, volume 5, pages 579–583, 2010.
- [225] P. Wark and J. Holt. A repeated matching heuristic for the vehicle routing problem. *Journal of Operational Research Society*, 45:1156–1167, 1994.

- [226] L. D. Whitley, V. S. Gordon, and K. E. Mathias. Lamarckian evolution, the baldwin effect and function optimization. In *Parallel problem solving from nature: PPSN III*, pages 6–15. Springer-Verlag, 1994.
- [227] T. Witkowski, S. Elzway, A. Antczak, and P. Antczak. *Advanced Intelligent Computing Theories and Applications*, chapter Representation of Solutions and Genetic Operators for Flexible Job Shop Problem. Communications in Computer and Information Science. Springer, 2007.
- [228] Y. G. Woldesenbet, G. G. Yen, and B. G. Tessema. Constraint handling in multiobjective evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 13(3):514–525, 2009.
- [229] D. H. Wolpert and W. G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.
- [230] Y. Wu, P. Ji, and T. Wang. An empirical study of a pure genetic algorithm to solve the capacitated vehicle routing problem. *ICIC Express Letters*, 2:41–45, 2008.
- [231] J. Xu and J. Kelly. A network flow-based tabu search for the vehicle routing problem. *Transportation Science*, 30:379–393, 1996.
- [232] S. Yang and C. Li. A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. *IEEE Transactions on Evolutionary Computation*, 14(6):959–974, 2010.
- [233] S. Yang and X. Yao. Population-based incremental learning with associative memory for dynamic environments. *IEEE Transactions on Evolutionary Computation*, 12(5):542–561, 2008.
- [234] W. Yang, K. Mathur, and R. H. Ballou. Stochastic vehicle routing problem with restocking. *Transportation Science*, 34(1):99 – 112, 2000.
- [235] X. S. Yang. *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, 2008.

- [236] T. Yoshida and H. Mori. Application of multi-objective memetic algorithm with solution diversity to probabilistic distribution network expansion planning. *IEEE Transactions on Power and Energy*, 129(12):1495–1502, 2009.
- [237] Q. Zhang and H. Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.
- [238] Q. Zhang, W. Liu, E. Tsang, and B. Virginas. Expensive multiobjective optimization by moea/d with gaussian process model. *IEEE Transactions on Evolutionary Computation*, 14(3):456–474, 2010.
- [239] Q. Zhang, J. Sun, G. Xiao, and E. Tsang. Evolutionary algorithms refining a heuristic: A hybrid method for shared-path protections in wdm networks under srlg constraints. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(1):51–61, 2007.
- [240] Z. Zhou, Y. S. Ong, P. B. Nair, A. J. Keane, and K. Y. Lum. Combining global and local surrogate models to accelerate evolutionary optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(1):66–76, 2007.

List of Related Publications

Journal

- **X. S. Chen**, “An algorithm development environment for problem-solving: software review”, *Memetic Computing Journal*, Vol. 4, No. 2, pp. 149-161, 2012.
- **X. S. Chen**, L. Feng and Y. S. Ong, “Self-Adaptive Memeplexes Robust Search Scheme for solving Stochastic Demands Vehicle Routing Problem”, *International Journal of Systems Science*, Vol. 43, No. 7, pp. 1347-1366, 2012.
- **X. S. Chen** and Y. S. Ong, “A Conceptual Modeling of Meme Complexes in Stochastic Search”, *IEEE Transactions on Systems, Man and Cybernetics, Part C*, Vol. 42, No. 3, 2012.
- **X. S. Chen**, Y. S. Ong, M. H. Lim and K. C. Tan, “A Multi-Facet Survey on Memetic Computation”, *IEEE Transactions on Evolutionary Computation*, Vol. 15, No. 5, pp. 591-607, 2011.
- **X. S. Chen**, Y. S. Ong, and M. H. Lim. “Cooperating Memes for Robust Vehicle Routing”, *International Journal of Innovative Computing, Information and Control*, Vol. 7, No. 11, pp. 6483-6506, 2011.
- Y. S. Ong, M. H. Lim and **X. S. Chen**, “Research Frontier: Memetic Computation - Past, Present & Future”, *IEEE Computational Intelligence Magazine*, Vol. 5, No. 2, pp. 24-36, 2010
- K. K. Lim, Y. S. Ong, M. H. Lim, **X. S. Chen**, and A. Agarwal. “Hybrid ant colony algorithms for path planning in sparse graphs”, *Soft Computing*, Vol. 12, No. 10, pp. 981-994, 2008.

Conference

- H.E. Huang, Y. S. Ong, C. S. Ho, **X. S. Chen**, “IGA Flock Brush for Non-Photorealistic Rendering”, *The 9th International Conference on Simulated Evolution and Learning*, Vol. 7673, pp. 480-490, 2012.
- L. Feng, Y. S. Ong, A. H. Tan and **X. S. Chen**, “Towards Human-like Social Multi-agents with Memetic Automaton”, *IEEE Congress on Evolutionary Computation*, pp. 1092-1099, 2011
- C. S. Ho, Q. H. Nguyen, Y. S. Ong, **X. S. Chen**: “Autonomous Multi-agents in Flexible Flock Formation”, *The Third international conference on Motion in games* , pp. 375-385, 2012
- **X. S. Chen**, M. H. Lim, and Y. S. Ong. “An Ant colony system algorithm for path planning in sparse graphs”, *International Conference on Intelligent & Advanced Systems 2007*, pp. 31-36, 2007.
- **X. S. Chen**, M. H. Lim, and D. C. Wunsch II. “A memetic algorithm configured via a problem solving environment for the hamiltonian cycle problems”, *IEEE Congress on Evolutionary Computation*, pp. 2766-2773, 2007.