8-2015

Memes as building blocks: a case study on evolutionary optimization + transfer learning for routing problems

Liang FENG

Yew-Soon ONG

Ah-hwee TAN
Singapore Management University, ahtan@smu.edu.sg

Ivor W. TSANG

Memetic Comp. (2015) 7:159–180
DOI 10.1007/s12293-015-0166-x

REGULAR RESEARCH PAPER

Memes as building blocks: a case study on evolutionary optimization + transfer learning for routing problems

Liang Feng[1] · **Yew-Soon Ong**[2] · **Ah-Hwee Tan**[2] · **Ivor W. Tsang**[3]

Received: 12 October 2014 / Accepted: 15 July 2015 / Published online: 5 August 2015

Abstract A significantly under-explored area of evolutionary optimization in the literature is the study of optimization methodologies that can evolve along with the problems solved. Particularly, present evolutionary optimization approaches generally start their search from scratch or the ground-zero state of knowledge, independent of how similar the given new problem of interest is to those optimized previously. There has thus been the apparent lack of automated knowledge transfers and reuse across problems. Taking this cue, this paper presents a *Memetic Computational Paradigm* based on *Evolutionary Optimization + Transfer Learning* for search, one that models how human solves problems, and embarks on a study towards intelligent evolutionary optimization of problems through the transfers of structured knowledge in the form of memes as building blocks learned from previous problem-solving experiences, to enhance future evolutionary searches. The proposed approach is composed of four culture-inspired operators, namely, Learning, Selection, Variation and Imitation. The role of the *learning* operator is to mine for latent knowledge buried in past experiences of problem-solving. The learning task is modelled as a mapping between past problem instances solved and the respective optimized solution by maximizing their statistical dependence. The *selection* operator serves to identify the high quality knowledge that shall replicate and transmit to future search, while the *variation* operator injects new innovations into the learned knowledge. The *imitation* operator, on the other hand, models the assimilation of innovated knowledge into the search. Studies on two separate established NP-hard problem domains and a realistic package collection/deliver problem are conducted to assess and validate the benefits of the proposed new memetic computation paradigm.

Keywords Memetic computation · Evolutionary optimization of problems · Learning from past experiences · Culture-inspired · Evolutionary learning · Transfer learning

✉ Liang Feng
liangf@cqu.edu.cn

Yew-Soon Ong
asysong@ntu.edu.sg

Ah-Hwee Tan
asahtan@ntu.edu.sg

Ivor W. Tsang
Ivor.Tsang@uts.edu.au

[1] College of Computer Science, Chongqing University, Chongqing, China

[2] Center for Computational Intelligence, School of Computer Engineering, Nanyang Technological University, Singapore, Singapore

[3] Centre for Quantum Computation and Intelligent Systems, University of Technology, Sydney, Australia

1 Introduction

Today, it is well recognized that the processes of learning and the transfer of what has been learned are central to humans in problem-solving [1]. Learning has been established to be fundamental to human in functioning and adapting to the fast evolving society. Besides learning from the successes and mistakes of the past and learning to avoid making the same mistakes again, the ability of human in selecting, generalizing and drawing upon what have been experienced and learned in one context, and extending them to new problems is deem to be most remarkable [2,3].

Within the context of computational intelligence, several core learning technologies in neural and cognitive systems, fuzzy systems, probabilistic and possibilistic reasoning have

been notable for their ability in emulating some of human's cultural and generalization capabilities [4–7], with many now used to enhance our daily life. Recently, in contrast to traditional machine learning approaches, *Transfer Learning* which uses data from a related source task to augment learning in a new or target task, has attracted extensive attentions and demonstrated great success in a wide range of real-world applications including computer vision, natural language processing, speech recognition, etc [8–13]. In spite of the accomplishments made in computational intelligence, the attempts to emulate the cultural intelligence of human in search, evolutionary optimization in particular, have to date received far less attention. In particular, existing evolutionary algorithms (EAs) have remained yet to fully exploit the useful traits that may exist in similar tasks or problems. In particular, the study of optimization methodology that evolves along with the problems solved has been under-explored in the context of evolutionary computation. To date, most evolutionary computation approaches continue to start a search on the given new problem of interest from ground zero state [14–23]. Thus a major gap of existing evolutionary search methodologies proposed in the literature is the lack of available techniques to enhance evolutionary search on related new problems by learning from past related solved problems.

In the literature, memetic computation has been defined as a paradigm that uses the notion of meme(s) as units of information encoded in computational representation for the purpose of problem-solving. The memes are captured from recurring information patterns or structures and can be evolved to form more complex higher level structures. However, currently, a meme has usually been perceived as a form of individual learning procedure, adaptive improvement procedure or local search operator to enhance the capability of population based search algorithm [24–26]. From the last decades, this integration has been established as an extension of the canonical evolutionary algorithm, by the names of hybrid, adaptive hybrid or Memetic Algorithm (MA) in the literature [25,27–30]. Falling back on the basic definition of a meme by Dawkins and Blackmore [31,32], as the fundamental building blocks of culture evolution, research on memetic computation can perhaps be more meme-centric focus by treating memes as the building blocks of a given problem domain.

In this paper, we embark on a study towards a new *Memetic Computation Paradigm*: *Evolutionary Optimization + Transfer Learning* for search, one that models how human solves problems. We believe that by leveraging from the potential common characteristics among the problem instances that belongs to the same problem domain, i.e., topological properties, data distributions or otherwise, the effective assessments of future unseen related problem instances can be achieved more efficiently, without the need to perform an exhaustive search each time or start the evolutionary search from a ground-zero knowledge state. Above and beyond the standard mechanisms of a conventional evolutionary search, for instance the genetic operators in the case of Genetic Algorithm, our proposed approach has four additional culture-inspired operators, namely, *Learning*, *Selection*, *Variation* and *Imitation*. The role of the *learning operator* is to mine for knowledge meme[1] from past experiences of problem-solving, which shall then manifest as instructions to bias the search on future problems intelligently (i.e., thus narrowing down the search space). The *selection operator*, on the other hand, selects the high quality knowledge meme that shall then replicate and undergo new innovations via the *variation operator*, before drawing upon them to enhance future evolutionary search. Last but not least, the *imitation operator* defines the assimilation of knowledge meme in subsequent problem solving. To summarize, the core contributions of the current work is multi-facets, which are outlined as follows:

1. To date, most search methods start the optimization process from scratch, with the assumption of zero usable information, i.e., ignoring how similar the current problem instance of interest is to those encountered in the past [14,34–36]. The current work endeavors to fill this gap by embarking a study on evolutionary optimization methodology with transfer capabilities that evolves along with the problems solved.
2. To the best of our knowledge, the present study serves as a first attempt to propose *transfer learning* as culture-inspired operators in the spirit of memetic computation (comprising of the mechanisms of cultural learning, selection, variation and imitation [27,30,33,36,37]), as a form of 'Intelligent Initialization' of high quality solutions in the starting population of the conventional evolutionary optimization so as to speed up future search on related problems.
3. Beyond the formalism of simple and adaptive hybrids as memetic algorithm, this paper introduces and showcases the novel representation of acquired knowledge from past optimization experiences in the form of memes. In contrast to the manifestation of memes as refinement procedures in hybrids, here memes manifest as natural building blocks of meaningful information, and in the present context, serving as the instructions for generating solutions that would lead towards optimized solutions[2] both efficiently and effectively.

[1] A meme is defined as the basic unit of cultural transmission in [31] stored in brains. In the context of computational intelligence, memes are defined as recurring real-world patterns or knowledge encoded in computational representations for the purpose of effective problem-solving [33].

[2] Optimized solution here denotes the best solution found by the evolutionary solvers.

4. We derive the mathematical formulations of the proposed transfer learning culture-inspired operators for faster evolutionary optimization of related problems. In this paper, we formulate the knowledge mining problem in the *learning operator* as a modelling of the mapping between past problem instances solved and the respective optimized solution via maximizing their statistical dependence. The *selection operator* is formulated as a maximization of the problem distributions similarity between instances, while *variation* and *imitation* are derived as the generalization of knowledge learned from past problem instances solved.
5. Comprehensive studies on two separate NP-hard problem domains using benchmark sets of diverse properties and a real world package collection/delivery problem showed that the proposed culture-inspired operators led to significant speedup in search performances and at no loss in solution quality, when incorporated into recently proposed evolutionary solvers. Notably, on several problem instances, improved search quality are observed over recently proposed state-of-the-art evolutionary solvers of the respective problem domains considered.

The rest of this paper is organized as follows: a brief discussion on the related works and the introduction of memes are given in Sect. 2.1. Section 3 introduces the proposed new memetic computation paradigm for search, via learning from past problem-solving experiences, to speedup evolutionary searches of related problems. Section 4 presents a brief discussion of the routing problem domain, particularly, capacitated vehicle routing problem (CVRP) and capacitated arc routing problem (CARP), and recently proposed evolutionary optimization methodologies for solving them. The proposed mathematical formulations and algorithms of the *learning*, *selection*, *variation* and *imitation* operators for fast evolutionary search on NP-hard routing problems are then described in Sect. 5. Last but not least, Sect. 6 presents and analyzes the detailed experimental results obtained on the CVRP and CARP benchmark sets and subsequently on a real world application. Lastly, the brief conclusive remarks of this paper are drawn in Sect. 7.

2 Preliminary

In this section, we first provide a brief review on related works in the literature. Subsequently, an overview of meme as building block of problems for problem-solving, which serves as one of the inspirations of this paper, is presented.

2.1 Related works

In practice, problems seldom exist in isolation, and previous related problem instances encountered often yield useful information that when properly harnessed, can lead to more efficient future evolutionary search. To date, some attempts have been made to reuse solutions from search experiences. Louis et al. [38], for instance, presented a study to acquire problem specific knowledge and subsequently using them to aid in the genetic algorithm (GA) search via case-based reasoning. Rather than starting anew on each problem, appropriate intermediate solutions drawn from similar problems that have been previously solved are periodically injected into the GA population. In a separate study, Cunningham and Smyth [39] also explored the reuse of established high quality schedules from past problems to bias the search on new traveling salesman problems (TSPs). Similar ideas on implicit and explicit memory schemes to store elite solutions have also been considered in dynamic optimization problems, where the objective function, design variables, and environmental conditions may change with time (for example, periodic changes) [40]. However, as both [38] and [39] as well as works on dynamic optimization problems [40] generally considered the exact storage of past solutions or partial-solutions from previous problems solved, and subsequently inserting them directly into the solution population of a new evolutionary search or the dynamic optimization search, they cannot apply well on unseen related problems that bear differences in structural properties, such as problem vertex size, topological structures, representations, etc. This means that what has been previously memorized from related problems solved cannot be directly injected into future search on unseen problems for successful reuse.

More recently, Pelikan et al. [41] proposed a framework to improve the model-directed optimization techniques by combining a pre-defined problem-specific distance metric with information mined from previous optimization experience on similar problems. They empirically illustrated the proposed approach can significantly speedup the original optimization technique. Further, Santana et al. [42] proposed to transfer the structural information from subproblems to bias the construction of aggregation matrix of the estimation of distribution algorithm (EDA) for solving multi-marker tagging single-nucleotide polymorphism (SNP) selection problem. They also obtained significant improvements over EDAs that do not incorporate information from related problems. However, it is worth noting that, since these transfer approaches are designed for model-based evolutionary optimization methods (e.g., EDA), they cannot apply with the model free evolutionary algorithms, such as genetic algorithm. Last but not least, Santana et al. [43] introduced to use network theory for mining structural information for evolutionary optimization, but how the mined information be used across

problems for enhancing evolutionary search was not discussed.

In contrast to existing approaches, the present new memetic computation paradigm addresses the task of learning generic building blocks or knowledge of useful traits from past problems solving experiences and subsequently drawing upon them through the cultural evolutionary mechanisms of learning, selection, variation and imitation (as opposed to a simple direct copying of past solutions or mode based approach in previous works) to speedup the search on new problems of the same domain. In such a manner, the transfer and incorporation of knowledge meme as generic building blocks of useful traits, can apply with model free evolutionary optimization and then lead to enhanced search on problems of differing vertex size, topological structures, and representations, etc.

2.2 Meme as building block of problems

Like gene in genetics, a meme is synonymous to memetic as being the building block of cultural know-how that is transmissible and replicable [30]. In the last decades, meme has inspired the new science of memetics which today represents the mind-universe analog to genetics in cultural evolution, stretching across the field of biology, cognition, psychology, and sociology [33].

Looking back on the history of meme, the term can be traced back to Dawkins [31] in his book "The selfish Gene", where he defined it as "a unit of information residing in the brain and is the replicator in human cultural evolution". Like genes that serve as "instructions for building proteins", memes are then "instructions for carrying out behavior, stored in brains". As discussed by Blackmore in her famous book "The Meme Machine", where she reaffirmed meme as information copied from one person to another and discussed on the theory of "memetic selection" as the survival of the fittest among competitive ideas down through generations [32]. Other definitions of meme that took flights from there have since emerged to include "memory item, or portion of an organism's neurally-stored information" [44], "unit of information in a mind whose existence influences events such that more copies of itself get created in other minds" [45], and "contagious information pattern that replicates by parasitically infecting human minds" [46].

In the literature, beyond the formalism of simple and adaptive hybrids in MA, Situngkir [47] presented a structured analysis of culture by means of memetics, where meme was regarded as the smallest unit of information. Heylighen and Chielens [48] discussed the replication, spread and reproduction operators of memes in cultural evolution. Nguyen et al. [49] studied the notion of "Universal Darwinism" and social memetics in search, and investigated on the transmission of memetic material via non-genetic means while Meuth et al. [50] proposed a new paradigm of meta-learning memetic computing for search. In their work, they demonstrated the concept of meta-learning with a memetic system, consisting of an optimizer, a memory, a selection mechanism, and a generalization mechanism that conceptualizes memes not just within the scope of a problem instance, but over a more generic contextual scope. More recently, Feng et al. [51] presented a memetic search with inter-domain Learning, wherein meme is defined as the knowledge building blocks that can be reused across different problem domains for enhanced evolutionary search.

3 Proposed memetic computation paradigm

In this section, we shall present the proposed memetic computation paradigm: evolutionary optimization + transfer learning. In particular, four culture-inspired operators, which introduce high quality solutions into the initial population of the evolutionary search on related problems, thus leading to enhanced optimization performances, are proposed. In our approach, the instructions for carrying out the behavior to act on a given problem are modeled as knowledge memes. The knowledge memes serve as the building blocks of past problems solving experiences that may be efficiently passed on or replicated to support the search on future unseen problems, by means of cultural evolution. This capacity to draw on the knowledge from previous instances of problem-solving sessions in the spirit of *memetic computation* [27,30,33] thus allows future search to be more efficient on related problems.

3.1 Transfer learning as culture-inspired operators

The proposed memetic computation paradigm based on evolutionary optimization (i.e., Fig. 1a) + transfer learning (i.e., Fig. 1b) is depicted in Fig. 1. In the figure, $\mathbf{P_{old}}$ is the set of past problems solved with $\mathbf{S_{old}}$ denoting the respective optimized solutions of $\mathbf{P_{old}}$. $\mathbf{P_{new}}$ is the set of unseen problems of interest to be optimized. And $\mathbf{S^j_{new}}$ is the initialized population of potential solutions for unseen problem $\mathbf{p^j}$. $\mathbf{M}$ denotes a knowledge meme. The proposed paradigm is composed of a conventional evolutionary algorithm as depicted in Fig. 1a (or it can be any state-of-the-art evolutionary algorithm in the domain of interest) and four culture-inspired operators proposed for facilitating faster evolutionary optimization of related problems as depicted in Fig. 1b, namely *Learning*, *Selection*, *Variation* and *Imitation*, whose functions are described in what follows:

– *Learning* operator: Given that $\mathbf{p}$ corresponds to a problem instance and $\mathbf{s}^*$ denotes the optimized solution of $\mathbf{p}$, as attained by an evolutionary solver (labeled here as *ES*).

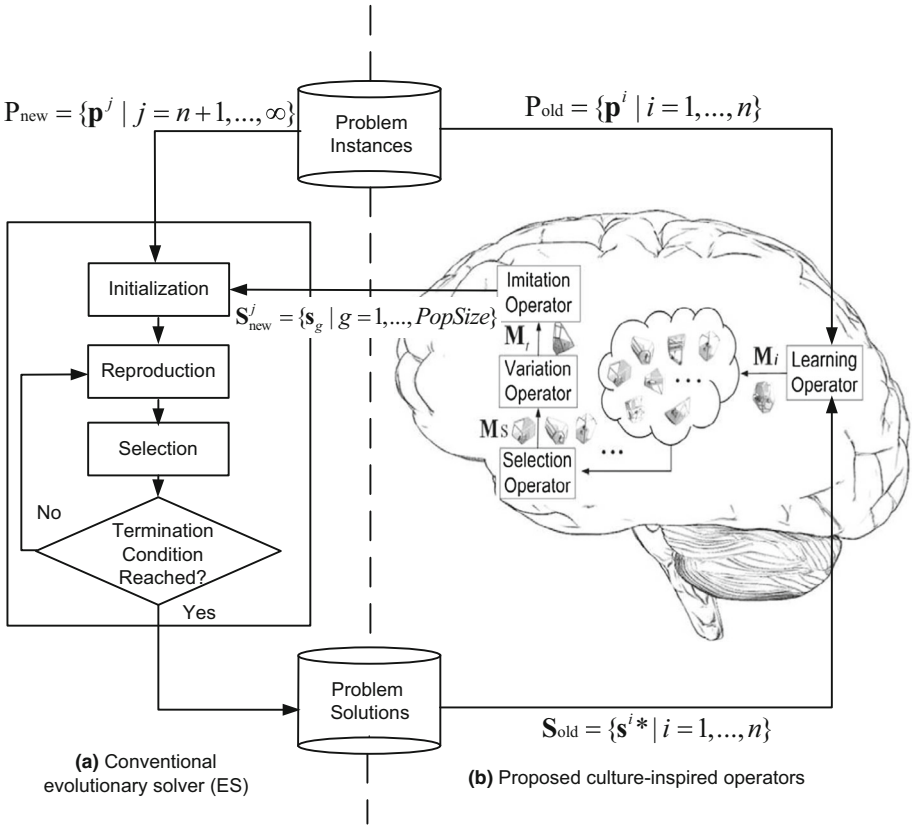


Fig. 1 Proposed memetic computation paradigm: evolutionary optimization (i.e., **a**) + transfer learning (i.e., **b**)

The learning operator takes the role of modeling the mapping from $\mathbf{p}$ to $\mathbf{s}^*$, to derive the knowledge memes. Thus, the learning process evolves in an incremental manner, and builds up the wealth of ideas in the form of identified knowledge, along with the number of problem instances solved. Note the contrast to a simple storage or exact memory of specific problem instance $\mathbf{p}$ with associated solution $\mathbf{s}^*$ as considered in the previous studies based on case-based reasoning [38].

– *Selection operator* Different prior knowledge introduces unique forms of bias into the search. Hence a certain bias would make the search more efficient on some classes of problem instances but not for others. Inappropriately harnessed knowledge, on the other hand, may lead to the possible impairments of the search. The selection operator thus serves to select the high quality knowledge, from the knowledge pool, that replicate successfully.
– *Variation operator* Variation forms the intrinsic innovation tendency of the cultural evolution. Without variations, maladaptive form of bias may be introduced in the evolutionary searches involving new problem instances. For instance, a piece of knowledge, which has been established as beneficial based on its particular demonstration of success on a given problem instance would quickly spiral out of control via replication. This will suppress the diversity and search of the evolutionary optimization across problems. Therefore, variation is clearly essential for retaining diversity in the knowledge pool towards efficient and effective evolutionary search.
– *Imitation operator* From Dawkins's book entitled "The selfish Gene" [31], ideas are copied from one person to another via imitation. In the present context, knowledge memes that are learned from past problem solving experiences replicate by means of imitation and used to enhance future evolutionary search on newly encountered problems.

3.2 Learning from past experiences

The schemata representation of knowledge meme in computing as the latent pattern is first identified. The problem solving experiences on the encountered problems are then captured via *learning* and crystallized as a part of the knowledge pool that form the memes or building blocks in the society of mind [52]. In this manner, whenever a new problem comes about, the *selection* operator kicks in to first identify the appropriate knowledge memes from the wealth of previously accumulated knowledge. These knowledge memes then undergo *variations* to effect the emergence of innovative knowledge. Enhancements to subsequent problem-solving efficiency on given new problems is then achieved by means of *imitation*.

Referring to Fig. 1, at time step $i = 1$, the evolutionary solver *ES* is faced with the first problem instance $\mathbf{p^1}$ to search on. Since $\mathbf{p^1}$ denotes the first problem of its kind to be optimized, no prior knowledge is available for enhancing the evolutionary solver, *ES*, search.[3] This is equivalent to the case where a child encounters a new problem of its kind to work on, in the absence of a priori knowledge that he/she could leverage upon. This condition is considered as "no relevant knowledge available" and the search by solver *ES* shall proceed normally, i.e., the *selection* operator remains dormant. If $\mathbf{s^{1*}}$ corresponds to the optimized solution attained by solver *ES* on problem instance $\mathbf{p^1}$ and $\mathbf{M}$ denotes the knowledge meme or building block, then $\mathbf{M_1}$ is the learned knowledge derived from $\mathbf{p^1}$ and $\mathbf{s^{1*}}$ via the *learning* operator. Since the learning process is conducted offline to the optimization process of future related problems, there is no additional computational burden placed on the existing evolutionary solver *ES*. On subsequent unseen problem instances $j = 2, \ldots, \infty$, *selection* kicks in to identify the appropriate knowledge memes **M**s from the knowledge pool, denoted here as **SoM**. Activated knowledge memes **M**s then undergo the *variation* operator to arrive at innovated knowledge $\mathbf{M_t}$ that can be *imitated* to bias subsequent evolutionary optimizations by the *ES*. In this manner, useful experiences attained from previously solved problem instances are captured incrementally and archived in knowledge pool **SoM** to form the society of mind, which are appropriately activated to enhance future search performances.

Like knowledge housed in the human mind for coping with our everyday life and problem solving, knowledge memes residing in the artificial mind of the evolutionary solver play the role of biasing the search positively on newly encountered problems. In this manner, the intellectual capacity of the evolutionary solver evolves along with the number of problems solved, with transferrable knowledge meme accumulating with time. When a new problem is encountered, suitable learned knowledge meme is activated and varied to guide the solver in the search process. This knowledge pool thus formed the evolving problem domain knowledge that may be activated to solve future evolutionary search efficiently.

4 Case studies on routing problems

In this section, we present the two widely studied challenging NP-hard domains on capacitated vehicle routing (CVR) and capacitated arc routing (CAR) considered in the present study.

[3] If a database of knowledge memes that are learned from relevant past problem solving experiences in the same domain is available, it can be loaded and leveraged upon.

4.1 Capacitated vehicle routing problem

The capacitated vehicle routing problem (CVRP) introduced by Dantzig and Ramser [53], is a problem to design a set of vehicle routes in which a fixed fleet of delivery vehicles of uniform capacity must service known customer demands for single commodity from a common depot at minimum cost. The CVRP can be formally defined as follows. Given a connected undirected graph $G = (V, E)$, where vertex set $V = \{v_i\}, i = 1 \ldots n$, n is the number of vertex, edge set $E = \{e_{ij}\}, i, j = 1 \ldots n$ denoting the arc between vertices v_i and v_j. Vertex v_d corresponds to the depot at which k homogeneous vehicles are based, and the remaining vertices denote the customers. Each arc e_{ij} is associated with a non-negative weight c_{ij} , which represents the travel distance from v_i to v_j. Consider a demand set $D = \{d(v_i)|v_i \in V\}$, where $d(v_i) > 0$ implies customer v_i requires servicing (i.e., known as task), the CVRP consists of designing a set of least cost vehicle routes $\mathcal{R} = \{\mathcal{C}_i\},\ i = 1 \ldots k$ such that

1. Each route $\mathcal{C}_i,\ i \in [1, k]$ must start and end at the depot node $v_d \in V$.
2. The total load of each route must be no more than the capacity W of each vehicle, $\sum_{\forall v_i \in \mathcal{C}} d(v_i) \leq W$.
3. $\forall v_i \in V \ and \ d(v_i) > 0$, there exists one and only one route $\mathcal{C}_i \in \mathcal{R}$ such that $v_i \in \mathcal{C}_i$.

The objective of the CVRP is to minimize the overall distance $cost(R)$ traveled by all k vehicles and is defined as:

$$cost(R) = \sum_{i=1}^{k} c(\mathcal{C}_i) \tag{1}$$

where $c(\mathcal{C}_i)$ is the summation of the travel distance e_{ij} contained in route $\mathcal{C}_i$.

4.2 Capacitated arc routing problem

The capacitated arc routing problem (CARP) was first proposed by Golden and Wong [54] in 1981. Instead of serving a set of customers (i.e., nodes, vertices) in CVRP, CARP is to serve a set of streets or segments. It can be formally stated as follows: Given a connected undirected graph $G = (V, E)$, where vertex set $V = \{v_i\}, i = 1 \ldots n$, n is the number of vertex, edge set $E = \{e_i\}, i = 1 \ldots m$ with m denoting the number of edges. Consider a demand set $D = \{d(e_i)|e_i \in E\}$, where $d(e_i) > 0$ implies edge e_i requires servicing (i.e., known as task), a travel cost vector $C_t = \{c_t(e_i)|e_i \in E\}$ with $c_t(e_i)$ representing the cost of traveling on edge e_i, a service cost vector $C_s = \{c_s(e_i)|e_i \in E\}$ with $c_s(e_i)$ representing the cost of servicing on edge e_i. A solution of CARP can be represented as a set of travel circuits $\mathcal{R} = \{\mathcal{C}_i\},\ i = 1 \ldots k$ which satisfies the following constraints:

1. Each travel circuit $\mathcal{C}_i$, $i \in [1, k]$ must start and end at the depot node $v_d \in V$.
2. The total load of each travel circuit must be no more than the capacity W of each vehicle, $\sum_{\forall e_i \in \mathcal{C}} d(e_i) \leq W$.
3. $\forall e_i \in E \; and \; d(e_i) > 0$, there exists one and only one circuit $\mathcal{C}_i \in \mathcal{R}$ such that $e_i \in \mathcal{C}_i$.

The cost of a travel circuit is then defined by the total service cost for all edges that needed service together with the total travel cost of the remaining edges that formed the circuit:

$$cost(\mathcal{C}) = \sum_{e_i \in \mathcal{C}_s} c_s(e_i) + \sum_{e_i \in \mathcal{C}_t} c_t(e_i) \tag{2}$$

where $\mathcal{C}_s$ and $\mathcal{C}_t$ are edge sets that required servicing and those that do not, respectively. And the objective of CARP is then to find a valid solution $\mathcal{R}$ that minimizes the total cost:

$$C_{\mathcal{R}} = \sum_{\forall \mathcal{C}_i \in \mathcal{R}} cost(\mathcal{C}_i) \tag{3}$$

4.3 Solving of CVRP and CARP domains

Here we generalized the solving of routing problems as searching for the suitable task assignments (i.e., vertices or arcs that require to be serviced) of each vehicle, and then finding the optimal service order of each vehicle for the assigned tasks. In the evolutionary search literature, the task assignment stage has been realized by means of simple task randomization [29] to more advance strategies including heuristic search, clustering [55], etc., while the optimal service order of each vehicle is attained via the mechanisms of evolutionary search operators. The example of an optimized routing solution can be illustrated in Fig. 2, where four vehicle routes, namely, $R_1 = \{0, \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, 0\}$, $R_2 = \{0, \mathbf{v}_6, \mathbf{v}_5, \mathbf{v}_4, 0\}$, $R_3 = \{0, \mathbf{v}_{10}, \mathbf{v}_9, \mathbf{v}_8, \mathbf{v}_7, 0\}$ and $R_4 = \{0, \mathbf{v}_{14}, \mathbf{v}_{13}, \mathbf{v}_{12}, \mathbf{v}_{11}, 0\}$, can be observed. A '0' index value is assigned at the beginning and end of route to denote that each route starts and ends at the depot.

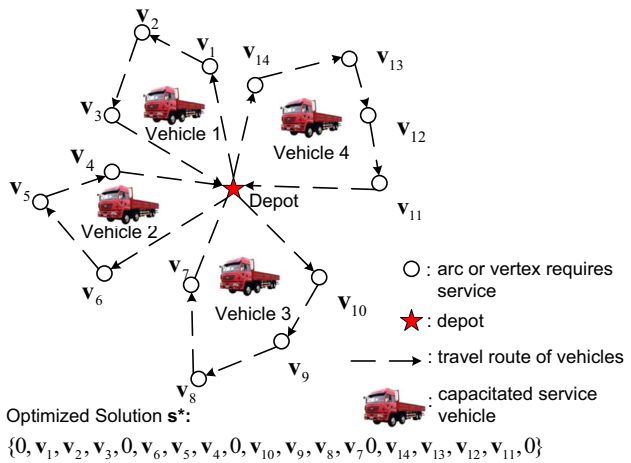


Fig. 2 The example of a CARP or CVRP

Theoretically, routing problems have been proven to be NP-hard with only explicit enumeration approaches known to solve them optimally. However, large scale problems are generally computationally intractable due to the poor scalability of most enumeration methods. From a survey of the literature, metaheuristics, heuristics and evolutionary computation have played important roles in algorithms capable of providing good solutions within tractable computational time. For CVRP, Cordeau et al. [56] considered a unified tabu search algorithm (UTSA) for solving VRP. Prins [57] presented an effective evolutionary algorithm with local search for the CVRP, while Reimann et al. [58] proposed a D-ants algorithm for CVRP which equipped ant colony algorithm with individual learning procedure. Recently, Lin et al. [59] takes the advantages of both simulated annealing and tabu search, and proposed a hybrid meta-heuristic algorithm for solving CVRP. Further, Chen et al. [55] proposed a domain-specific cooperative memetic algorithm for solving CVRP and achieved better or competitive results compared with a number of state-of-the-art memetic algorithms and meta-heuristics to date.

On the other hand, for CARP, Lacomme et al. in [60] presented the basic components that have been embedded into memetic algorithms (MAs) for solving the extended version of CARP (ECARP). Lacomme's MA (LMA) was demonstrated to outperform all known heuristics on three sets of benchmarks. Recently, Mei et al. [61] extended Lacomme's work by introducing two new local search methods, which successfully improved the solution qualities of LMA. In a separate study, a memetic algorithm with extended neighborhood search was also proposed for CARP in [29]. Further, Liang et al. proposed a formal probabilistic memetic algorithm for solving CARP, with new best-known solutions to date found on 9 of the benchmark problems [62].

In what follows, we present the formulations of the proposed four culture-inspired operators in the context of routing problems for learning and transferring useful traits from past problem solving experiences as knowledge meme, that can be used to enhanced future routing search process.

5 Proposed formulations and algorithms: CVRPs & CARPs

In this section, we present the proposed formulation and algorithmic implementations of the transfer learning culture-inspired operators, namely, *learning*, *selection*, *variation* and *imitation* for faster evolutionary optimization of related problems in two domains described in Sect. 4, namely CVRPs and CARPs.

Algorithm 1: Pseudo code of Fast Evolutionary Optimization of CVRPs/CARPs by Transfer Learning from Past Experiences.

Begin:
for $j = 1 : \infty$ *new problem instances* $\mathbf{p}_{new}^{j}$ *or* $\mathbf{X}_{new}^{j}$ **do**
  if SoM! $= \emptyset$ **then**
    /*knowledge pool not empty*/
    Perform *selection* to identify high quality knowledge memes or distance matrices $\mathbf{M}$s $\in$ **SoM** /*see Eqn. 12 in later Sect. 5.2*/
    Perform *variation* to derive generalized knowledge $\mathbf{M_t}$ from $\mathbf{M}$s.
    Perform *imitation* of $\mathbf{M_t}$ on $\mathbf{X}_{new}^{j}$ to derive the transformed problem distribution $\mathbf{X}_{new}^{j'}$, where
      $\mathbf{X}_{new}^{j'} = Transform(\mathbf{X}_{new}^{j}, \mathbf{M_t})$
    Empty the initial solution population Ω.
    for $g = 1 : Population\ Size$ **do**
      /*Fig. 5(a)→Fig. 5(b)*/
      1. $Task\ Assignment\ of\ \mathbf{s}_g =$
        $KMeans(\mathbf{X}_{new}^{j'}, Vehicle\ No., RI)$
      /Fig. 5(b)→Fig. 5(c), RI denotes random initial points*/
      2. $Service\ Order\ of\ \mathbf{s}_g = PDS(\mathbf{X}_{new}^{j'})$
      /*Fig. 5(c)→Fig. 5(d), $PDS(\cdot)$ denotes the pairwise distance sorting*/
      3. Insert $\mathbf{s}_g$ into Ω.
  else
    Proceed with the original population initialization scheme of the evolutionary solver *ES*.
  /*Start of Evolutionary Solver *ES* Search*/
  Perform reproduction and selection operations of *ES* with generated population $\mathbf{s}_g$ until the predefined stopping criteria are satisfied.
  /*End of Evolutionary Solver ES Search*/
  Perform *learning* on given $\mathbf{p}_{new}^{j}$ and corresponding optimized solution $\mathbf{s}_{new}^{j*}$ denoted by $(\mathbf{X}_{new}^{j}, \mathbf{Y}_{new}^{j})$, attained by *ES* evolutionary solver to derive knowledge $\mathbf{M}_{new}^{j}$.
  Archive the learned knowledge of $\mathbf{p}_{new}^{j}$ into **SoM** knowledge pool for subsequent reuse.
End

The pseudo-code and detailed workflow for the realizations of the proposed 'fast evolutionary optimization by transfer learning from past experiences' on CVRP or CARP problem domains, are outlined in Algorithm. 1. For a given new routing problem instance $\mathbf{p}_{new}^{j}$ (with data representation $\mathbf{X}_{new}^{j}$) posed to evolutionary solver *ES*, the mechanisms of the *selection* operator kicks in to select the high quality knowledge memes $\mathbf{M}$s to activate, if the knowledge pool **SoM** is not empty. *Variation*, which takes inspirations from the human's ability to simplify from past knowledge learned in previous problem solving experiences, then operates on the activated knowledge memes $\mathbf{M}$s to arrive at the generalized knowledge $\mathbf{M_t}$. Subsequently, for given new problem instances $\mathbf{X}_{new}^{j}$, *imitation* proceeds to positively bias the search of evolutionary optimization solver *ES*, using the generalized knowledge $\mathbf{M_t}$, followed by clustering and pairwise distance sorting (*PDS*) to generate the biased tasks assignment and service orders solutions that would enhance the search performances on $\mathbf{p}_{new}^{j}$. When the search on $\mathbf{p}_{new}^{j}$ completes, the problem instance $\mathbf{p}_{new}^{j}$ together with the attained optimized solution $\mathbf{s}_{new}^{j*}$ of *ES*, i.e., $\mathbf{X}_{new}^{j}$ and $\mathbf{Y}_{new}^{j}$ which denote the matrix representation of $\mathbf{p}_{new}^{j}$ and $\mathbf{s}_{new}^{j*}$ (see Fig. 4), respectively, then undergo the *learning* operation so as to update the knowledge pool **SoM**.[4]

5.1 Learning operator

This subsection describes the learning of knowledge memes, as building blocks of useful traits from given routing problem instances $\mathbf{p}$ and the corresponding optimized solutions $\mathbf{s}^*$ (i.e., Line 25 in Algorithm 1). To begin, we refer to Fig. 2, which shall serve as the example routing problem instance used in our illustrations. Figure 3 on the other hand illustrate the learning of knowledge $\mathbf{M}$ from an optimized routing problem instance and subsequently using this knowledge to bias the tasks assignment and ordering of a routing problem. Specifically, Fig. 3a depicts the distribution of the tasks in the example routing problem of Fig. 2 that need to be serviced. Figure 3b then denotes the optimized routing solution of the *ES* evolutionary solver on problem Fig. 2 or 3a. The dashed circles in Fig. 3b denote the task assignments of the individual vehicles and the arrows indicate the tasks service orders, as optimized by *ES*.

Here a knowledge meme $\mathbf{M}$ is defined in the form of a distance matrix that maximally aligns the given original distribution and service orders of tasks to the optimized routing solution $\mathbf{s}^*$ attained by solver *ES*. Using the example routing problem instance in Fig. 2, we formulate the knowledge meme as matrix $\mathbf{M}$ that transforms or maps the task distributions depicted in Fig. 3a to the desired tasks distribution of $\mathbf{s}^*$ while preserving the corresponding tasks service orders, as depicted in Fig. 3b. In this manner, whenever a new routing problem instance is encountered, suitable learned knowledge memes from previously optimized problem instances is then deployed to realign the tasks distribution and service orders constructively. For instance, Fig. 3c showcases the desirable scaled or transformed tasks distribution of Fig. 3a when the appropriate knowledge meme $\mathbf{M}$ is put to work. In particular, it can be observed in Fig. 3c that we seek for the knowledge memes necessary to re-locate tasks serviced by a common vehicle to become closer to one another (as desired by the optimized solution $\mathbf{s}^*$ shown in Fig. 3b), while tasks serviced

[4] Note that as the learning operation is conducted offline, it does not incur additional cost to the evolutionary optimization of $\mathbf{p}_{new}^{j}$.

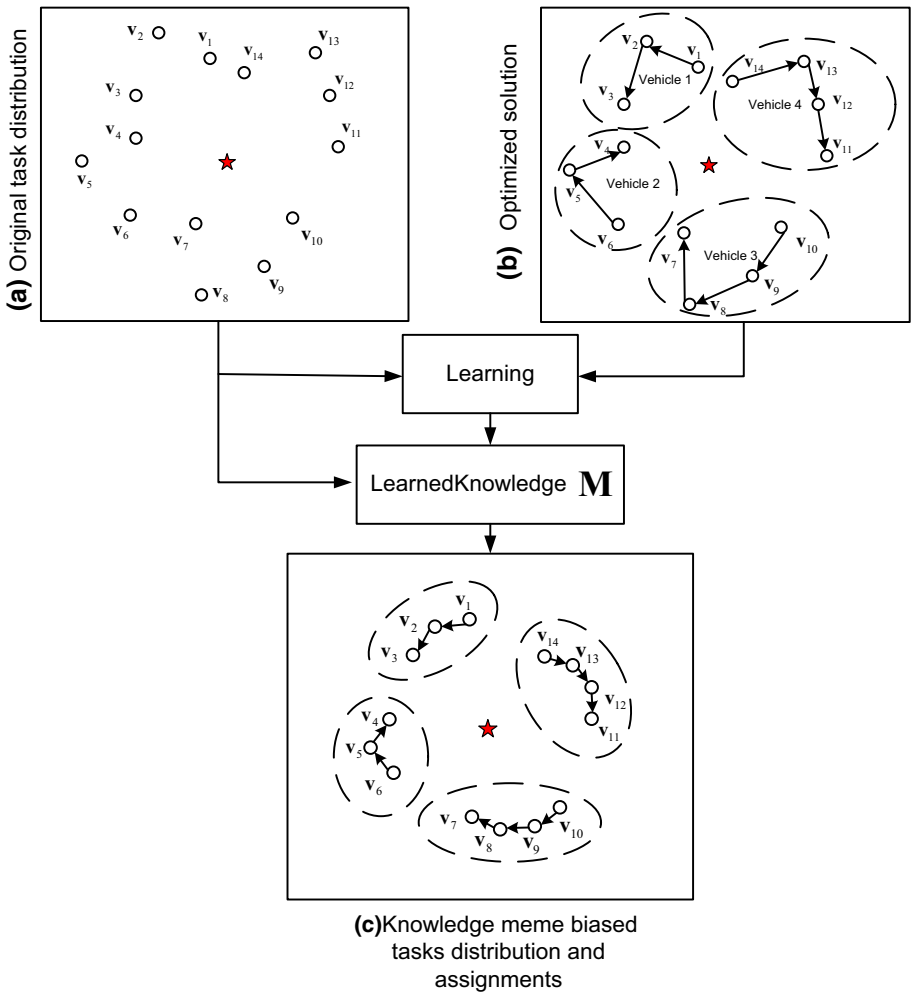


Fig. 3 Learning of knowledge **M** which shall serve as the instruction for biasing the tasks assignment and ordering of a routing problem

by different vehicles to be mapped further apart. Further, to match the service orders of each vehicle to that of the optimized solution $\mathbf{s}^*$, the task distribution is adapted according to the sorted pairwise distances in ascending order (e.g., the distance between v_1 and v_3 is the largest among v_1, v_2 and v_3, while the distance between v_{10} and v_9 is smaller than that of v_{10} and v_8).

In what follows, the proposed mathematical definitions of a knowledge meme **M** for the transformations of tasks distribution are detailed. In particular, given $\mathbf{V} = \{\mathbf{v}_i \,|i = 1, \ldots, n\}$, n is the number of tasks, denoting the tasks of a problem instance to be assigned. The distance between any two tasks $\mathbf{v}_i = (v_{i1}, \ldots, v_{ip})^T$ and $\mathbf{v}_j = (v_{j1}, \ldots, v_{jp})^T$ in the p-dimensional space $\mathbb{R}^p$ is then given by:

$$d_M(\mathbf{v}_i, \mathbf{v}_j) = ||\mathbf{v}_i - \mathbf{v}_j||_M = \sqrt{(\mathbf{v}_i - \mathbf{v}_j)^T \mathbf{M} (\mathbf{v}_i - \mathbf{v}_j)}$$

where T denotes the transpose of a matrix or vector. **M** is positive semidefinite, and can be represented as $\mathbf{M} = \mathbf{L}\mathbf{L}^T$ by means of singular value decomposition (SVD). Substituting this decomposition into $d_M(\mathbf{v}_i, \mathbf{v}_j)$, we arrive at:

$$d_M(\mathbf{v}_i, \mathbf{v}_j) = \sqrt{(\mathbf{L}^T\mathbf{v}_i - \mathbf{L}^T\mathbf{v}_j)^T (\mathbf{L}^T\mathbf{v}_i - \mathbf{L}^T\mathbf{v}_j)} \quad (4)$$

From Eq. 4, it is worth noting that the distances among the tasks are scaled by meme **M**. Thus we derive at a knowledge meme **M** that performs the realignment of tasks distribution and service orders of a given new problem instance to one that bears greater similarity to the optimized solution $\mathbf{s}^*$.

Next, the proposed mathematical formulations for learning of knowledge meme **M** are given. The schemata representations of a problem instance (**p**), optimized solution ($\mathbf{s}^*$) and distance constraints set $\mathcal{N}$ are first defined. In particular, the data representations of the example problem instance in Fig. 2 is depicted in Fig. 4, where v_{11}, v_{12}, etc., denote the features representation of each task, and $\mathrm{D}(\cdot)$ indicates the Euclidean distance metric. Further, if task $\mathbf{v}_i$ and task $\mathbf{v}_j$ are served by the same vehicle, $\mathbf{Y}(i, j) = 1$, otherwise, $\mathbf{Y}(i, j) = -1$. The distance constraints set $\mathcal{N}$ contains the service order information of the tasks and derived from the optimized solution $\mathbf{s}^*$. With respect to the example in Fig. 2, since task $\mathbf{v}_3$ is served after $\mathbf{v}_2$ from $\mathbf{v}_1$, the constraint thus takes the form of $\mathrm{D}(\mathbf{v}_1, \mathbf{v}_3) > \mathrm{D}(\mathbf{v}_1, \mathbf{v}_2)$ as depicted in Fig. 4.

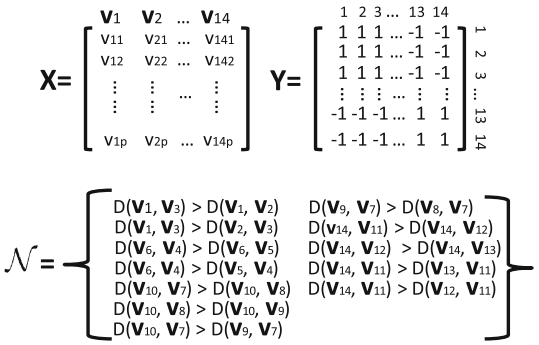


Fig. 4 Data representations of a problem instance $\mathbf{p} = \mathbf{X}$, the corresponding optimized solution $\mathbf{s}^* = \mathbf{Y}$ and distance constraints set $\mathcal{N}$

To derive the knowledge meme $\mathbf{M}$ of a given CVRP or CARP problem instance, denoted by $(\mathbf{p}, \mathbf{s}^*)$, we formulate the learning task as a maximization of the statistical dependency[5] between $\mathbf{X}$ and $\mathbf{Y}$ with distance constraints as follows:

$$\begin{aligned} &\max_{\mathbf{K}} tr(\mathbf{HKHY}) \\ &\text{s.t.} \mathbf{K} = \mathbf{X}^T * \mathbf{M} * \mathbf{X} \\ &\quad D_{ij} > D_{iq}, \forall (i,j,q) \in \mathcal{N}, \quad \mathbf{K} \succeq 0 \end{aligned} \tag{5}$$

where $tr(\cdot)$ denotes the trace operation of a matrix. $\mathbf{X}$, $\mathbf{Y}$ are the matrix representations of a CARP or CVRP instance $\mathbf{p}$ and the corresponding problem solution $\mathbf{s}^*$, respectively.

Further, $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}'$ centers the data and the labels in the feature space, $\mathbf{I}$ denotes the identity matrix, n equals to the number of tasks. $D_{ij} > D_{iq}$ is then the constraint to impose a vehicle to serve task q before task j, upon serving task i.

Let $\mathbf{T}_{ij}$ denotes a $n \times n$ matrix that takes non-zeros at $T_{ii} = T_{jj} = 1$, $T_{ij} = T_{ji} = -1$. The distance constraints $D_{ij} > D_{iq}$ in Eq. 5 is then reformulated as $tr(\mathbf{KT}_{ij}) > tr(\mathbf{KT}_{iq})$. Further, slack variables ξ_{ijq} are introduced to measure the violations of distance constraints and penalize the corresponding square loss. Consequently, by substituting the constraints into Eq. 5, we arrive at:

$$\begin{aligned} &\min_{\mathbf{M},\xi} \; -tr(\mathbf{XHYHX}^T\mathbf{M}) + \frac{C}{2}\sum \xi_{ijq}^2 \\ &\text{s.t.} \;\; \mathbf{M} \succeq 0 \\ &\quad tr(\mathbf{X}^T\mathbf{MXT}_{ij}) > tr(\mathbf{X}^T\mathbf{MXT}_{iq}) - \xi_{ijq}, \\ &\quad \forall (i,j,q) \in \mathcal{N} \end{aligned} \tag{6}$$

where C balances between the two parts of the criterion. The first constraint enforces the learnt knowledge denoted by matrix $\mathbf{M}$ to be positive semi-definite, while the second constraint imposes the scaled distances among the tasks to align well with the desired service orders of the optimized solution $\mathbf{s}^*$ (i.e., $\mathbf{Y}$).

To solve the learning problem in Eq. 6, we first derive the minimax optimization problem by introducing dual variables α for the inequality constraints based on Lagrangian theory.

$$\begin{aligned} Lr = &tr(-\mathbf{HX}^T\mathbf{MXHY}) + \frac{C}{2}\sum \xi_{ijq}^2 \\ &- \sum \alpha_{ijq}(tr(\mathbf{X}^T\mathbf{MXT}_{ij}) - tr(\mathbf{X}^T\mathbf{MXT}_{iq}) + \xi_{ijq}) \end{aligned} \tag{7}$$

Set $\frac{\partial Lr}{\partial \xi_{ijq}} = 0$, we have:

$$C\sum \xi_{ijq} - \sum \alpha_{ijq} = 0 \Longrightarrow \xi_{ijq} = \frac{1}{C}\sum \alpha_{ijq} \tag{8}$$

By substituting Eq. 8 into Eq. 7, we reformulate the learning problem in Eq. 6 as a minimax optimization problem, which is given by:

$$\begin{aligned} \max_{\alpha}\min_{\mathbf{M}} \; &tr\left[\left(-\mathbf{XHYHX}^T - \sum \alpha_{ijq}\mathbf{XT}_{ij}\mathbf{X}^T \right.\right. \\ &\left.\left. + \sum \alpha_{ijq}\mathbf{XT}_{iq}\mathbf{X}^T\right)\mathbf{M}\right] - \frac{1}{2C}\sum \alpha_{ijq}^2 \\ &\text{s.t.} \;\; \mathbf{M} \succeq 0 \end{aligned} \tag{9}$$

By setting

$$\mathbf{A} = \mathbf{XHYHX}^T + \sum \alpha_{ijq}\mathbf{XT}_{ij}\mathbf{X}^T - \sum \alpha_{ijq}\mathbf{XT}_{iq}\mathbf{X}^T \tag{10}$$

and

$$\Delta J_{ijq}^t = tr[(\mathbf{XT}_{iq}\mathbf{X}^T - \mathbf{XT}_{ij}\mathbf{X}^T)\mathbf{M}] - \frac{1}{C}\alpha_{ijq} \tag{11}$$

Upon solving the above formulations and derivations, we arrive at Eqs. 10 and 11. Then, as a common practice in Machine Learning, parameter C of Eq. 11 is configured by means of cross validation and the learning problem of Eq. 9 is solved using readily available methods [64]. As the update of $\mathbf{M}$ is the SVD on $\mathbf{A}$, its complexity is $O(p^3)$, where p is the dimension. The complexity of calculating $\mathbf{A}$ is $O(n^2p + np^2 + mp^2)$, where n is the number of vertices, and m is the number of constraints in $\mathcal{N}$. Further, the complexity of updating each α_{ijq} is $O(p^2r)$, where r is the rank of $\mathbf{M}$. Thus the complexity of updating all α is $O(mp^2r)$.

[5] Dependency is a measure of the correlation of two random variables [63]. Here our interest on knowledge meme $\mathbf{M}$ is in the form of a maximization of the statistical dependency, so as to ensure a maximal alignment between the transformed tasks distribution and the tasks distribution of the optimized solution. The trace of $\mathbf{HKHY}$ is the empirical estimation of HSIC criterion [63], which is a nonlinear statistical dependence measures defined on two sets of random variables $\mathbf{X}$ and $\mathbf{Y}$, in their feature spaces, $\phi(\mathbf{X})$ and $\psi(\mathbf{Y})$. Mathematically, it tries to measure $||C_{\mathbf{XY}}||^2$, where $C_{\mathbf{XY}} := E_{\mathbf{X},\mathbf{Y}}[(\phi(\mathbf{X}) - \mu_{\mathbf{X}}) \otimes (\psi(\mathbf{Y}) - \mu_{\mathbf{Y}})]$, $\mu_{\mathbf{X}}$ and $\mu_{\mathbf{Y}}$ are the mean measures of $\phi(\mathbf{X})$ and $\psi(\mathbf{Y})$. A higher HSIC thus implies a higher nonlinear dependence between X and Y in the sense of the $\phi(\mathbf{X})$ and $\psi(\mathbf{Y})$ feature spaces.

5.2 Selection operator

Since different knowledge memes introduces unique biases into the evolutionary search, inappropriately chosen knowledge and hence biases can lead to potential negative impairments of the evolutionary search. To facilitate a positive transfer of knowledge that would lead to enhanced evolutionary search, the *selection* operator (i.e., Line 5 in Algorithm 1) is designed to select and replicate from high quality knowledge memes that share common characteristics with the given new problem instance of interest. In particular, for a given set of z unique **M**s in **SoM**, i.e., $\mathbf{SoM} = \{\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_z\}$ that form the knowledge pool, the *selection* operator is designed to give higher the weights μ_i to knowledge memes that would induce positive biases. Further, as we consider the learning and selection of knowledge memes from problems of a common domain, positive correlation among the problems is a plausible assumption.[6] If the unseen problem instance is very different from the past problems solved (e.g., the customer distribution of the new unseen problem differs greatly from previously solved problems in the CVRP and CARP), the selection operator shall not deploy any inappropriate knowledge memes, since no positive knowledge existed in the past experiences. In this case, the original state-of the-art optimization solver shall operate as routine.

Here, the knowledge meme coefficient vector $\boldsymbol{\mu}$ is derived based on the maximum mean discrepancy criterion.[7]

$$\begin{aligned} &\max_{\mu,\mathbf{Y}} tr(\mathbf{H}\mathbf{X}^T\mathbf{M}_t\mathbf{X}\mathbf{H}\mathbf{Y}) + \sum_{i=1}^{z}(\mu_i)^2 Sim_i \\ &\text{s.t.}\mathbf{M}_t = \sum_{i=1}^{z}\mu_i\mathbf{M}_i, \quad \mathbf{M}_i \succeq 0 \\ &\qquad \mu_i \geq 0, \quad \sum_{i=1}^{n}\mu_i = 1 \end{aligned} \tag{12}$$

In Eq. 12, the first term serves to maximize the statistical dependence between input matrix **X** and output label **Y** of the clusters of tasks. The second term measures the similarity between previous problem instances solved to the given new problem of interest. Sim_i defines the similarity measure between two given problem instances. In vehicle routing, tasks distribution and vehicle capacity are two key features that define the problem. Hence the similarity measure is formulated here as $Sim_i = -(\beta * MMD_i + (1-\beta) * DVC_i)$, where $MMD(D_s, D_t) = ||\frac{1}{n_s}\sum_{i=1}^{s}\phi(x_i^s) - \frac{1}{n_t}\sum_{i=1}^{t}\phi(x_i^t)||$ with $\phi(\mathbf{x}) = \mathbf{x}$ denoting the maximum mean discrepancy between the distribution of two given instances by considering the distance between their corresponding means. D_s and D_t are the two given routing problem instances, with x_i^s and x_i^t denoting the location information of a customer in D_s and D_t, respectively. DVC_i denotes the discrepancy in vehicle capacity between any two problem instances. The vehicle capacity is available as part of the problem definition. From domain knowledge, the task distribution (location of nodes to be serviced) has a higher weightage than vehicle capacity information. This implies that $\beta > 0.5$. In this work, β is configured empirically as 0.8 to favour task distribution information over vehicle capacity information.

In Eq. 12, two unknown variables exist (i.e., μ and **Y**). **Y** is obtained from the results of task assignment (i.e., if task $\mathbf{v}_i$ and task $\mathbf{v}_j$ are served by the same vehicle, $\mathbf{Y}(i, j) = 1$, otherwise, $\mathbf{Y}(i, j) = -1$. The respective task assignment is obtained by clustering on the **M** transformed tasks **X**.). With **Y** fixed, Equation 12 becomes a quadric programming problem of μ. To solve the optimization problem of Eq. 12, we first perform clustering (e.g., K-Means) [65] on input **X** directly to obtain the label matrix **Y**. By keeping **Y** fixed, we obtained μ by maximizing Eq. 12 via quadric programming solver. Next, by maintaining the chosen **M** fixed, clustering is made on the new $\mathbf{X}^{'}$ (i.e., transformed by selected **M**. $\mathbf{X}' = \mathbf{L}^T\mathbf{X}$, where **L** is obtained by SVD on **M**) to obtain label matrix **Y**.

5.3 Variation operator

Further, to introduce innovations into the selected knowledge memes during subsequent reuse, the *variation* operator (i.e., Line 6 in Algorithm 1) kicks in. In the present context, we take inspirations from human's ability to generalize from past problem solving experiences. Hence *variation* is realized here in the form of *generalization*. However, it is worth noting that other alternative forms of probabilistic scheme in variations may also be considered since uncertainties can generate growth and variations of knowledge that we have of the world [66], hence leading to higher adaptivity capabilities for solving complex and non-trivial problems.

Here, the *variation* is derived as a generalization of the selected knowledge memes:

$$\mathbf{M}_t = \sum_{i=1}^{z}\mu_i\mathbf{M}_i, \left(\sum_{i=1}^{z}\mu_i = 1, \mu_i \in [0, 1]\right) \tag{13}$$

where $\mathbf{M}_i$ denotes the meme knowledge stored in the meme pool, z is the total number of memes stored, and μ_i is obtained by *selection operator* discussed in Sect. 5.2.

5.4 Imitation operator

In CVRP and CARP, the search for optimal solution is typically solved as two separate phases. The first phase involves

[6] From our experimental study, the problems in the benchmark set are mostly verified to be positively correlated.

[7] Maximum mean discrepancy measures the distribution differences between two data sets, which can come in the form of vectors, sequences, graphs, and other common structured data types.

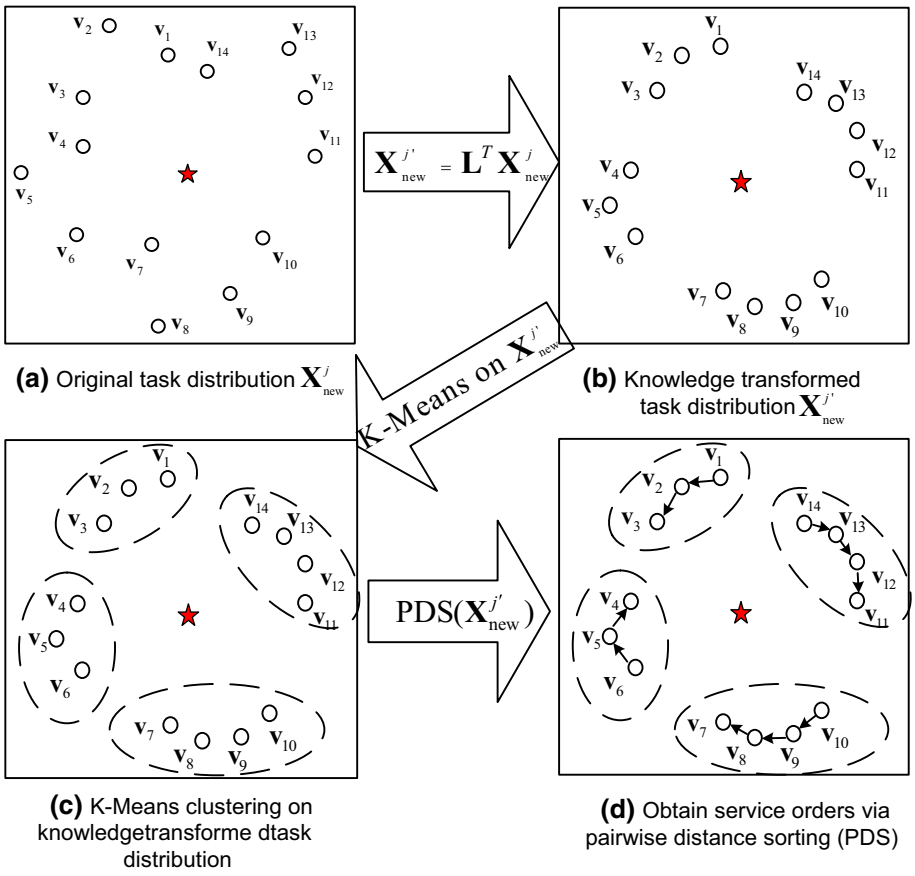


Fig. 5 An illustration of knowledge imitation in the generating positively biased CVRP or CARP solutions

the assignment of the tasks that require services to the appropriate vehicles. The second phase then serves to find the optimal service order of each vehicle for the assigned tasks obtained in phase 1.

In what follows, the imitation of learned knowledge memes to bias the initial population of solutions in subsequent *ES* searches are described. For each solution $\mathbf{s}_g$ in the EA population, the knowledge $\mathbf{M}_t$ generalized from past experiences is imitated for the purpose of generating positively biased solutions (see Line 10 in Algorithm 1) in the evolutionary search by transforming or remapping the original tasks distribution solution (i.e., both tasks assignments and tasks service orders), as denoted by $\mathbf{X}^{j}_{new}$, to become new tasks distribution $\mathbf{X}^{j'}_{new}$ given by:

$$\mathbf{X}^{j'}_{new} = \mathbf{L}^T \mathbf{X}^{j}_{new} \tag{14}$$

where $\mathbf{L}$ is derived by singular value decomposition of $\mathbf{M}_t$. An illustrative example is depicted in Fig. 5, where Fig. 5a denote the original task distribution $\mathbf{X}^{j}_{new}$, while Fig. 5b is the resultant knowledge biased or transformed tasks distribution $\mathbf{X}^{j'}_{new}$ using $\mathbf{M}_t$.

In phase 1, K-Means clustering with random initializations is conducted on the knowledge biased tasks distribution $\mathbf{X}^{j'}_{new}$ to derive the tasks assignments of the vehicles as depicted in Fig. 5c, where the dashed circles denote the task assignments of the individual vehicles, i.e., denoting the tasks that shall be serviced by a common vehicle.

In phase 2, the service orders of each vehicle are subsequently achieved by sorting the pairwise distances among tasks in an ascending order. The two tasks with largest distance shall then denote the first and last tasks to be serviced. Taking the first task as reference, the service order of the remaining tasks are defined according to the sorted orders. Referring to Fig. 5d as an example, where the arrows indicate the service orders of the tasks, the distance between $\mathbf{v}_{10}$ and $\mathbf{v}_7$ are the largest among $\mathbf{v}_{10}$, $\mathbf{v}_9$, $\mathbf{v}_8$ and $\mathbf{v}_7$. In assigning $\mathbf{v}_{10}$ as the reference task to be served, $\mathbf{v}_9$ is then the next task to be serviced, since the distance between $\mathbf{v}_{10}$ and $\mathbf{v}_9$ is smaller than that of $\mathbf{v}_{10}$ versus $\mathbf{v}_8$ or versus $\mathbf{v}_7$.

6 Experimental study

To evaluate the performance of the proposed transfer learning as culture-inspired operators for fast evolutionary optimization of related problems via transfer learning from past problem solving experiences, comprehensive empirical studies with regard to search speed and search solution quality, are conducted on the two challenging NP-hard routing problems

Table 1 Criteria for measuring performance

Criterion	Definition
Number of Fitness Evaluation	Average number of fitness evaluation calls made across all 30 independent runs conducted
Ave.Cost	Average travel cost or fitness of the solutions obtained across all 30 independent runs conducted
B.Cost	Best travel cost or fitness of the solutions obtained across all 30 independent runs conducted
Std.Dev	Standard deviation of the solutions' travel cost or fitness across all 30 independent runs conducted

and a real world application in this section. In particular, we first present studies on the capacitated vehicle routing problem (CVRP) domain and then the capacitated arc routing problem (CARP) domain. These consist of problem instances of diverse properties in terms of vertex size, topological structures (task distribution), and vehicle capacity, which cannot be readily handled using existing case based reasoning approaches [38,39] as previously discussed in Sect. 2.1. In the present study, two recently proposed evolutionary algorithms for solving CVRPs and CARPs, labeled in their respective published works as *CAMA* [67] and *ILMA* [61] respectively, are considered here as the baseline conventional evolutionary solvers for the independent domains. Further, several criteria defined to measure the search performances are then listed in Table 1. Among these criteria, *Number of Fitness Evaluation* is used to measure the efficiency of the algorithms, while *Ave.Cost* and *B.Cost* serve as the criteria for measuring the solution qualities of the algorithms.

6.1 Capacitated vehicle routing problem domain

6.1.1 Experimental configuration

All three commonly used CVRP benchmark sets with diversity properties (e.g., number of vertex, vehicle number, etc.) are investigated in the present empirical study, namely "AUGERAT", "CE" and "CHRISTOFIDES". The detailed properties (e.g., number of vertices, lower bound, etc.) of the CVRP instances considered are summarized in Tables 2 and 3, where V denotes the number of vertices that need to be serviced, C_v gives the capacity of the vehicles in each instance, and LB describes the lower bound of each problem instance. In CVRP, each task or vertex has a corresponding *coordinates* (i.e., 2-d space) and *demand*. Using the *coordinates* of the vertex, the tasks assignment of each vehicle are generated based on K-Means clustering. A CVRP instance is thus represented by input matrix $\mathbf{X}$, see Fig. 4, which is composed of the coordinate features of all tasks in the problem instance. The desired vehicle assigned for each task (i.e., task assignment) is then given by the *ES* optimized solution, $\mathbf{Y}$, of the respective CVRP instances.

Besides the proposed knowledge meme biased approach, two other commonly used initialization procedures for generating the population of solution individuals in the state-of-the-art baseline *CAMA* are investigated here to verify the efficiency and effectiveness of the proposed evolutionary search across problems. The first is the simple random approach for generating the initial population, which is labeled here as *CAMA-R*. The second is the informed heuristic population initialization procedure proposed in the state-of-the-art baseline *CAMA* [67] method. In particular, the initial population is a fusion of solution generated by *Backward Sweep*, *Saving*, and *Forward Sweep* and random initialization approaches.

The *CAMA* that employs our proposed transfer learning culture-inspired operators is then notated as *CAMA-M*, where the initial population of individuals in CAMA are now generated based on the high quality knowledge memes that have been accumulated from past CVRP solving experiences via the cultural evolutionary mechanisms of the *learning*, *selection*, *variation* and *imitation*. Note that if no prior knowledge has been learned so far, *CAMA-M* shall behave exactly like the baseline *CAMA*.

Last but not the least, the operator and parameter settings of *CAMA-R*, *CAMA*, *CAMA-M* are kept the same as that of [67] for the purpose of fair comparison. For *CAMA-M*, the MMD of Eq. 12 is augmented with the demand of each task as one of the problem feature.

Table 2 Properties of the "Augerat" CVRP benchmark set

CVRP instance	A-n32-k5	A-n54-k7	A-n60-k9	A-n69-k9	A-n80-k10	B-n41-k6	B-n57-k7	B-n63-k10	B-n68-k9	B-n78-k10	P-n50-k7	P-n76-k5
Number of vertices	31	53	59	68	79	40	56	62	67	77	49	75
Capacity of vehicle	100	100	100	100	100	100	100	100	100	100	150	280
Lower bound	784	1167	1354	1159	1763	829	1140	1496	1272	1221	554	627

Table 3 Properties of the "CE" and "Christofides" CVRP benchmark sets

CVRP instance	E-n33-k4	E-n76-k7	E-n76-k8	E-n76-k10	E-n76-k14	E-n101-k8	c50	c75	c100	c100b	c120	c150	c199
Number of vertices	32	75	75	75	75	100	50	75	100	100	120	150	199
Capacity of vehicle	8000	220	180	140	100	200	160	140	200	200	200	200	200
Lower bound	835	682	735	830	1021	815	524.61	835.26	826.14	819.56	1042.11	1028.42	1291.45

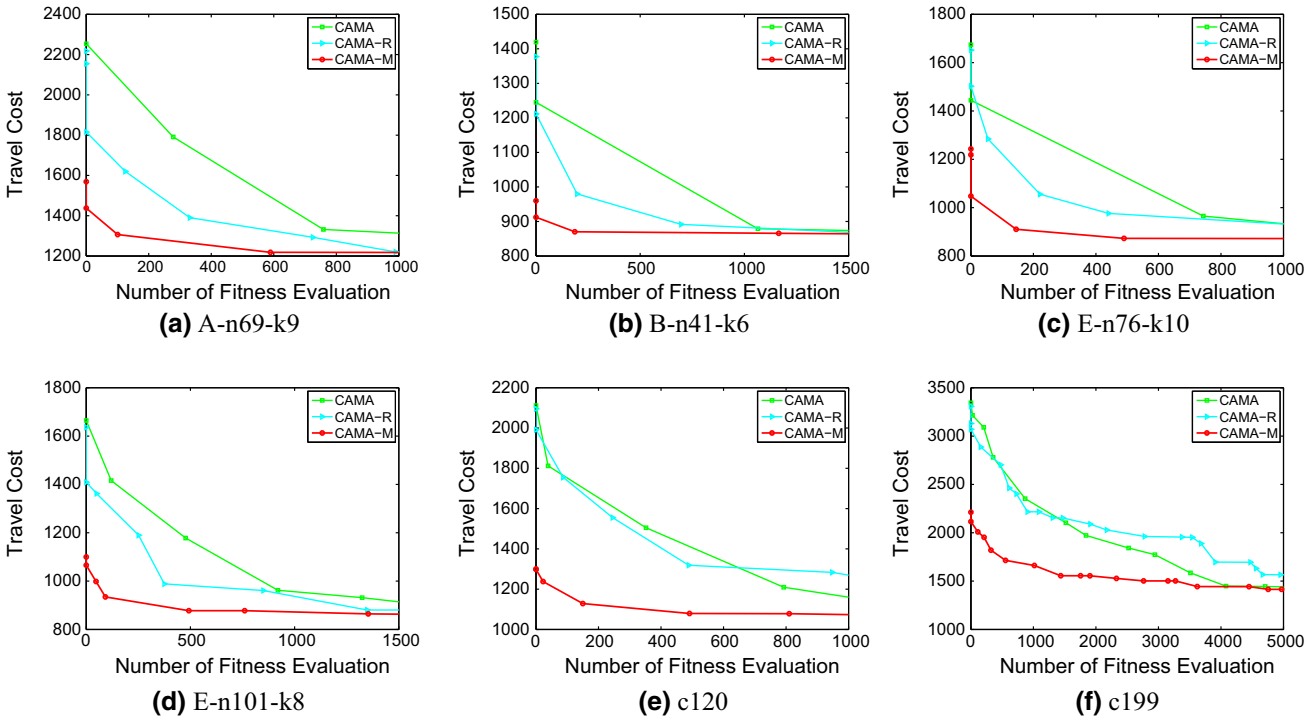


Fig. 6 Averaged search convergence traces (across 30 independent runs) of *CAMA*, *CAMA-R*, and *CAMA-M* on representative CVRP "AUGERAT", "CE", and "CHRISTOFIDES" benchmark sets. *Y*-axis: Travel cost, *X*-axis: Number of Fitness Evaluation. Note that *CAMA-M* is observed to search significantly faster in converging to near the lower bound solution on the respective CVRPs than the other counterparts

6.1.2 Results and discussions

To gain a better understanding on the performances of the proposed new memetic computation paradigm, we present, analyze, discuss and compare the results obtained against recently proposed methods based on the criteria of search efficiency and solution quality.

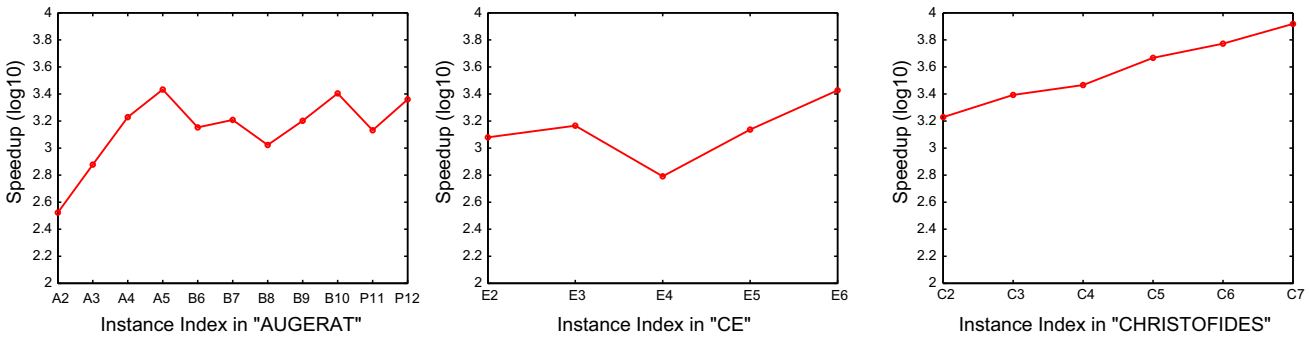
Search efficiency: convergence trends and speedup To assess the efficiency of the proposed approach, the representative search convergence traces of *CAMA*, *CAMA-R* and *CAMA-M* on the 3 different CVRP benchmark sets are presented in Fig. 6.[8] The *Y*-axis of the figures denote the *Actual travel cost* obtained, while the *X*-axis gives the respective computational effort incurred in terms of the *Number of Fitness Evaluation Calls* made so far. From these figures, it can be observed that *CAMA-M* converges rapidly to near the lower bound solution at very early stage of the search as compared to both *CAMA-R* and *CAMA* across all the CVRP instances. This is because of the high quality solutions introduced into the initial population of *CAMA-M* by the positive memes learned from past search experiences. For example, on instances "B-n41-k6" (Fig. 6b), *CAMA-M* takes only approximately 250 number of fitness evaluations to converge near to the lower bound solution while *CAMA-R* and *CAMA* incurred more than 1000 number of fitness evaluations to do so. On the larger problem instances, such as "c199" (Fig. 6f), the fitness evaluations savings is more sig-

[8] Due to page limit constraints, only representatives of each series have been shown.

Table 4 Speedup by *CAMA-M* over *CAMA* in the different stages of the search on representative CVRP instances has been observed

Benchmark set	Instances	*Speedup* (*Fitness*)							
		*Bin*1	*Bin*2	*Bin*3	*Bin*4	*Bin*5	*Bin*6	*Bin*7	*Bin*8
AUGERAT	A-n60-k9	754.55	16.89	10.87	10.16	15.75	16.52	9.72	1.75
		(1746.00)	(1697.38)	(1648.75)	(1600.13)	(1551.50)	(1502.88)	(1454.25)	(1405.63)
	B-n57-k7	1615.15	591.14	459.65	32.86	16.28	10.79	6.98	3.90
		(1361.00)	(1333.38)	(1305.75)	(1278.13)	(1250.50)	(1222.88)	(1195.25)	(1167.63)
	P-n50-k7	1354.55	124.08	61.44	17.96	17.68	17.47	3.36	1.22
		(684.00)	(668.13)	(652.25)	(636.38)	(620.50)	(604.63)	(588.75)	(572.88)
CE	E-n76-k10	618.18	13.62	8.32	2.27	2.26	6.51	11.43	6.11
		(1243.00)	(1192.38)	(1141.75)	(1091.13)	(1040.50)	(989.88)	(939.25)	(888.63)
CHRISTOFIDES	c100b	2924.24	31.81	17.46	18.42	19.81	21.77	26.95	9.44
		(1087.60)	(1054.15)	(1020.70)	(987.25)	(953.80)	(920.35)	(886.90)	(853.45)

The values embraced in brackets (.) denotes the actual fitness values for fitness bins 1–8



(a) AUGERAT **(b)** CE **(c)** CHRISTOFIDES

Fig. 7 Speedup of *CAMA-M* over *CAMA* across all the problem instances for the three CVRP benchmark sets. *Y*-axis: Speedup, *X*-axis: Problem Instance index of each CVRP in the benchmark set. A learning curve with increasing knowledge learned in each benchmark set, as observed from the increasing $log_{10}(Speedup)$ observed, as more problems are solved

nificant, where *CAMA-M* is observed to bring about at least 1000 fitness evaluations cost savings to arrive at the solution qualities attained by *CAMA* and *CAMA-R*.

To provide more in-depth insights on the enhanced efficiency of the *CAMA-M*, in Table 4, we also tabulated the amount of speedup by *CAMA-M* over the baseline *CAMA* in arriving at different stages of the search defined by the fitness levels, for the representative problem instances.[9] Here speedup is defined by $\frac{CAMA^i_{Fitness\ EvaluationCalls}}{CAMA-M^i_{Fitness\ EvaluationCalls}}$, where $i = 1...i...N$ and N denoting the number of fitness bins considered. $A^i_{Fitness\ EvaluationCalls}$ denotes the number of fitness evaluation used by algorithm A to arrive at the fitness attained in bin i. In the results reported in Table 4, an equal-width fitness bin size of 8 is used. For example, the *fitness bin* 1 and *bin* 8 of problem instance *c100b* in Table 4 (i.e., see the last row) shows the speedup of *CAMA-M* over *CAMA* at the start of the search and upon search convergence are 2924.24 and 9.44 times, respectively. Note that speedups are observed throughout the entire search in the representative CVRP instances, as observed in the table.

For the purpose of conciseness, we further summarize the $log_{10}(Speedup)$ of *CAMA-M* over *CAMA* at the start of the search on the CVRP instances in the order that they are solved, in Fig. 7. It is worth noting that Fig. 7 resembles a learning curve where the increasing knowledge learned corresponds to an increasing $log_{10}(Speedup)$ observed as more problems are solved in each benchmark set. For example, on the benchmark "AUGERAT" set, the $log_{10}(Speedup)$ is observed to increase from under 2.6 to exceed 3.4 when instances "A2", "A3" and "A4" are solved. Overall, a $log_{10}(Speedup)$ of at least 2.5 times has been

[9] Similar trends on enhancements in search efficiency of *CAMA-M* over *CAMA* has been obtained on all the other problem instances. However, due to the page limit constraint, only representatives of instances in each benchmark problem class can be presented in this paper.

Table 5 Solution quality of *CAMA*, *CAMA-R*, and *CAMA-M* on "AUGERAT", "CE" and "CHRISTOFIDES" CVRP benchmark sets

CVRP	*CAMA*			*CAMA-R*			*CAMA-M*		(Proposed method)
instance	*B.Cost*	*Ave.Cost*	*Std.Dev*	*B.Cost*	*Ave.Cost*	*Std.Dev*	*B.Cost*	*Ave.Cost*	*Std.Dev*
A1.A-n32-k5	784	748	0	784	784	0	784	784	0
A2.A-n54-k7	1167	1169.50	3.36	1167	**1167**	0	1167	**1167**+	0
A3.A-n60-k9	1354	1356.73	3.59	1354	1355.20	1.86	1354	**1354.4**+	1.22
A4.A-n69-k9	1159	1164.17	3.07	1159	1162.20	2.41	1159	**1161.43**+	2.37
A5.A-n80-k10	1763	1778.73	9.30	1763	1777.07	7.94	1763	**1775.7**≈	8.80
B6.B-n41-k6	829	**829.30**	0.47	829	829.93	0.94	829	829.53≈	0.73
B7.B-n57-k7	1140	1140	0	1140	1140	0	1140	1140≈	0
B8.B-n63-k10	1537	1537.27	1.46	**1496**	1528.77	15.77	**1496**	**1525.86**+	17.45
B9.B-n68-k9	1274	1281.47	5.56	1274	1284.80	4.51	**1273**	**1281.43**≈	5.74
B10.B-n78-k10	1221	1226.07	5.48	1221	1226.80	6.39	1221	**1224.37**≈	3.23
P11.P-n50-k7	554	556.33	2.34	554	554.93	1.72	554	**554.26**+	1.01
P12.P-n76-k5	627	630.70	5.34	627	628.87	1.61	627	**628.63**≈	1.51
E1.E-n33-k4	835	835	0	835	835	0	835	835≈	0
E2.E-n76-k7	682	685.67	2.17	682	684.73	1.31	682	**684.66**≈	1.12
E3.E-n76-k8	735	737.57	2.36	735	737.17	1.60	735	**737.06**≈	1.91
E4.E-n76-k10	830	837.03	3.56	831	835.80	3.19	830	**834.73**+	2.92
E5.E-n76-k14	1021	**1025.67**	3.48	1021	1026.27	3.33	1021	1025.80≈	3.64
E6.E-n101-k8	816	820.63	3.20	815	818.97	1.94	815	**818.53**+	1.55
C1.c50	524.61	525.45	2.56	524.61	**524.61**	0	524.61	525.73≈	2.90
C2.c75	835.26	842.32	4.04	835.26	840.28	3.52	835.26	**839.53**+	3.45
C3.c100	826.14	829.43	2.39	826.14	829.73	2.08	826.14	**829.13**≈	1.94
C4.c100b	819.56	819.56	0	819.56	819.56	0	819.56	819.56≈	0
C5.c120	1042.11	1044.18	2.21	1042.11	1043.08	1.13	1042.11	**1042.83**+	0.94
C6.c150	1032.50	1043.27	5.67	1034.19	1044.73	5.87	1030.67	**1041.97**≈	6.27
C7.c199	**1304.87**	**1321.17**	5.98	1313.46	1325.48	5.99	1308.92	1322.18≈	7.51
No. Win	1	3		1	2		**2**	**17**	

The superior solution quality of each respective problem instance is highlighted in bold font. "No. Win" denotes the number of instances that an algorithm achieved best performance. Note that *CAMA-M* is superior on 68 % (17/25) of the instances ("≈", "+" and "−" denote *CAMA-M* statistically significant similar, better, and worse than *CAMA*, respectively)

attained by *CAMA-M* on all the CVRP instances considered.

Solution Quality: To evaluate the solution quality of the proposed approach, Table 5 tabulates all the results obtained by respective algorithms over 30 independent runs. The values in "*B.Cost*" and "*Ave.Cost*" denoting superior performance are highlighted using bold font. Further, in order to obtain the statistically comparison, Wilcoxon rank sum test with 95 % confidence level has been conducted on the experimental results. As discussed in Sect. 3, the "knowledge pool" of *CAMA-M* is empty when the first CVRP instance is encountered (e.g., "A-n32-k5" of "AUGERAT" benchmark set), and thus *CAMA-M* behaves like the baseline *CAMA*. As more CVRP instances are encountered, the *learning*, *selection*, *variation* and *imitation* mechanisms shall kick in to learn and generalize knowledge that would induce positive biases into the evolutionary search of new CVRP instances. It can be observed from Table 5 that the *CAMA-M* converges to competitive solution qualities attained by both *CAMA-R* and *CAMA* on the first problem instance of each CVRP benchmarks (since no knowledge meme is learned yet), while exhibiting superior performances over *CAMA-R* and *CAMA* on subsequent CVRP instances. *Thus, beyond showing speedups in search performance at no loss in solution quality,* CAMA-M *has been observed to attain improved solution quality on the CVRPs*. In particular, on "AUGERAT" and "CE" benchmark sets, *CAMA-M* exhibits superior performances in terms of *Ave.Cost* on 13 out of 19 CVRP instances. In addition, on "CHRISTOFIDES" benchmark set, *CAMA-M* also attained improved solution quality in terms of *Ave.Cost* on 4 out of 7 CVRP instances. Since *CAMA*, *CAMA-R* and *CAMA-M* shares a common baseline evolutionary solver, i.e., *CAMA*, and differing only in terms of the population initialization phase, the superior performance

Table 6 Properties of the *egl* "E" Series CARP benchmarks

Data set	"E" Series											
	E1A	E1B	E1C	E2A	E2B	E2C	E3A	E3B	E3C	E4A	E4B	E4C
V	77	77	77	77	77	77	77	77	77	77	77	77
E_r	51	51	51	72	72	72	87	87	87	98	98	98
E	98	98	98	98	98	98	98	98	98	98	98	98
LB	3548	4498	5566	5018	6305	8243	5898	7704	10,163	6048	8884	11,427

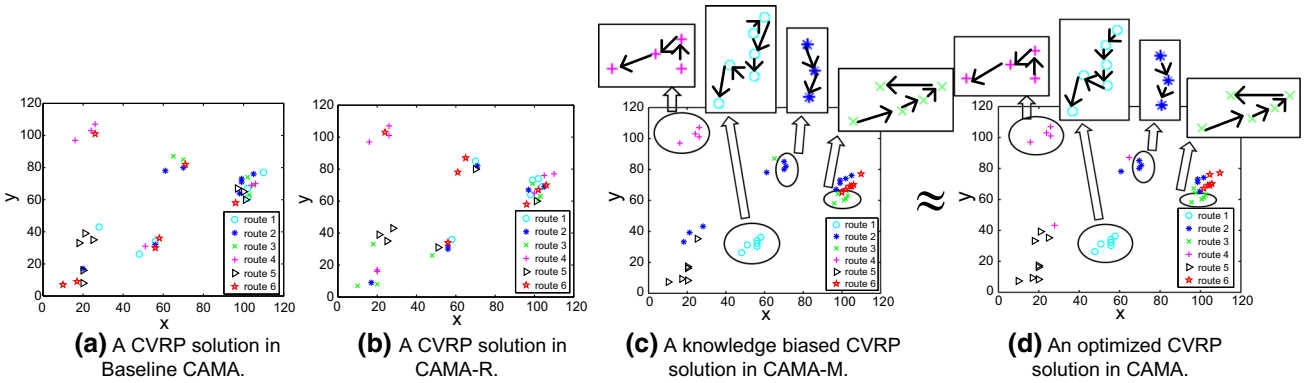


Fig. 8 An illustration of CVRP solutions in the respective EA populations for solving "B-n41-k6" CVRP instance. Each point plotted in the sub-figures denotes a CVRP customer node that needs service. Points or nodes with same symbol are serviced by a common vehicle

of *CAMA-M* can clearly be attributed to the effectiveness of the proposed transfer learning as culture-inspired operators where imitation of learned knowledge meme from past problem solving experiences are used to generate biased solutions that lead to enhanced future evolutionary searches.

Insights on a knowledge biased CVRP solution In this subsection, to provide a deep insight into the mechanisms of the proposed approach in attaining the high performance efficacy observed, we analyze samples of the solutions obtained by *CAMA*, *CAMA-R* and *CAMA-M*, as well as the converged optimized solution of *CAMA* on solving problem instance "B-n41-k6". In Fig. 8, each node denotes a customer that needs to be serviced, and the nodes with the same color and shape shall be serviced by a common route or vehicle. Figure 8a, b denote the solution in the initial population of baseline *CAMA* and *CAMA-R*, respectively. Figure 8c is the solution in *CAMA-M*, which has been positively biased using the imitated knowledge learned from past experiences of problem solving on CVRP instances "A-n32-k5", "A-n54-k7", "A-n60-k9" and "A-n69-k9". Further, Fig. 8d gives the converged optimized solution achieved by *CAMA*. As observed, the task distributions of the solution in *CAMA-M* search is noted to bear greatest similarities to that of the converged optimized solution by *CAMA*, as compared to that of *CAMA* and *CAMA-R*. Besides task distributions, portions of the figures are magnified in Fig. 8c, d, for the purpose of illustrating the service orders of a solution obtained by *CAMA-M* relative to the converged optimized solution of baseline *CAMA*, respectively. The magnified subfigures illustrate high similarities between their respective service orders.

This suggests that the service orders information of the converged optimized solution for instances "A-n32-k5", "An54-k7", "A-n60-k9" and "A-n69-k9" has been successful learned and preserved by the *learning* operator of the *CAMA-M*, and subsequently through the cultural evolutionary mechanisms of *selection*, *variation* and *imitation*, the learned knowledge meme is imitated to generate positively biased solutions that is close to the optimal solution, thus bringing about significant speedups in the search on related problem instances.

6.2 Capacitated arc routing problem

To assess the generality of the proposed transfer learning culture-inspired operators for faster evolutionary optimization of problems by learning from past experiences, further experimental study on the domain of capacitated arc routing problems is conducted in what follows.

The well-established *egl* benchmark set is used in the present experimental study on CARP. It comprises of two series of CARP instances, namely "E" and "S" series with a total of 24 instances. The detailed properties of each *egl* instance are presented in Tables 6 and 7. "$|V|$", "$|E_R|$", "E" and "LB" denote the number of vertices, number of tasks, total number of edges and lower bound, of each problem instance, respectively.

Table 7 Properties of the *egl* "S" Series CARP benchmarks

Data set	"S" Series											
	S1A	S1B	S1C	S2A	S2B	S2C	S3A	S3B	S3C	S4A	S4B	S4C
V	140	140	140	140	140	140	140	140	140	140	140	140
E_r	75	75	75	147	147	147	159	159	159	190	190	190
E	190	190	190	190	190	190	190	190	190	190	190	190
LB	5018	6384	8493	9824	12,968	16,353	10,143	13,616	17,100	12,143	16,093	20,375

In CARP, each task (arc) is represented by a corresponding *head vertex*, *tail vertex*, *travel cost* and *demand (service cost)*. Note the contrast to CVRP where a task is represented by a vertex. Thus the arc information is pre-processed to obtain the position vertices. The shortest distance matrix of the vertices is first derived by means of the Dijkstra's algorithm [68], i.e., using the distances available between the vertices of a CARP. The coordinate features (i.e., locations) of each task are then approximated by means of multidimensional scaling [69]. In this manner, each task is represented as a node in the form of coordinates and the dimension of the nodes is governed based on multidimensional scaling. A CARP instance in the current setting is then represented by input vector **X** composing of the coordinate features of all tasks in the problem. With such a representation, standard clustering approaches such as the K-Means algorithm is then used on the CARP for task assignments, followed by pairwise distances sorting of tasks to preserve the service orders. The label information of each task in **Y** belonging to the CARP instance, is then defined by the optimized solution of CARP.

In the empirical study, two commonly used population initialization procedures based on *ILMA* are considered here for comparison. The first is a simple random approach, which is labeled here as *ILMA-R*. The second is the informed heuristic based population generation procedure used in the baseline *ILMA* [61] for CARP. There, the initial population is formed by a fusion of chromosomes generated from Augment_Merge, Path_Scanning, Ulusoy's Heuristic and the simple random initialization procedures.

Our proposed transfer learning culture-inspired operators, which leverage past problem solving experiences to bias the initial or starting population of EA solutions in the conventional ILMA is labeled then here as *ILMA-M*. To facilitate a fair comparison and verify the benefits of the proposed transfer learning culture-inspired approach, the configurations of the evolutionary operators in baseline *ILMA* and the variants are keep in consistent to those reported in [61].

6.2.1 Results and discussions

On both the "E" and "S" series *egl* instances, *ILMA-M* has been consistently been able to converge more efficiently than *ILMA* and *ILMA-R* on the CARP instances considered.[10] Similar trends on speedup in search as observed in the CVRP has been observed in the CARP. Overall, *ILMA-M* is noted to bring up to 70 % savings in the number of fitness function evaluations to arrive at the solutions attained by both *ILMA* and *ILMA-R* on most of the CARP instances. For instance, it is worth noting that on instance "S1-B" (Fig. 9b), *ILMA-M* incurred a total of 1.5×10^6 number of fitness function evaluations to converge at the same solution found by *ILMA* and *ILMA-R*, which otherwise expended a large fitness evaluation costs of approximately 6×10^6. This equates to a significant cost savings of 4 times by *ILMA-M* over *ILMA* and *ILMA-R*.

Further, Table 8 summarizes the results that measure the solution quality on the "E"-Series and "S"-Series *egl* instances as obtained by the respective algorithms, across 30 independent runs. To obtain statistical significance comparisons, the Wilcoxon rank sum test with 5 % confidence level has been conducted on the experimental results. As observed, *ILMA-M* performed competitively to *ILMA* on instances "E1-A" and "S1-A", which is expected, since these are the first encountered problem instances of *ILMA-M* on the "E" and "S" *egl* CARP instances, respectively, where no useful prior knowledge is available yet. As more problem instances are solved, the *ILMA-M* is observed to not only search much more efficiently but also at no loss in solution quality. As a matter of fact, it arrives at superior solution quality over *ILMA* in terms of $Ave.Cost$ on 18 out of 24 problem instances in the *egl* benchmark set. In terms of $B.Cost$, *ILMA-M* also achieved 2 and 8 improved solution qualities over *ILMA* and *ILMA-R* on the "E" and "S" series *egl* instances, respectively.

6.3 Real world application: the package collection/delivery problem

In the courier business, the package collection/delivery task is among one of the challenging tasks that courier companies confront with everyday. The package collection/delivery problem (PCP) can be defined as the task of servicing a set of customers with a fleet of capacity constrained vehicles located at a single or multiple depot(s). In particular, the PCP

[10] Due to page limit constraints, only representatives of each series have been shown.

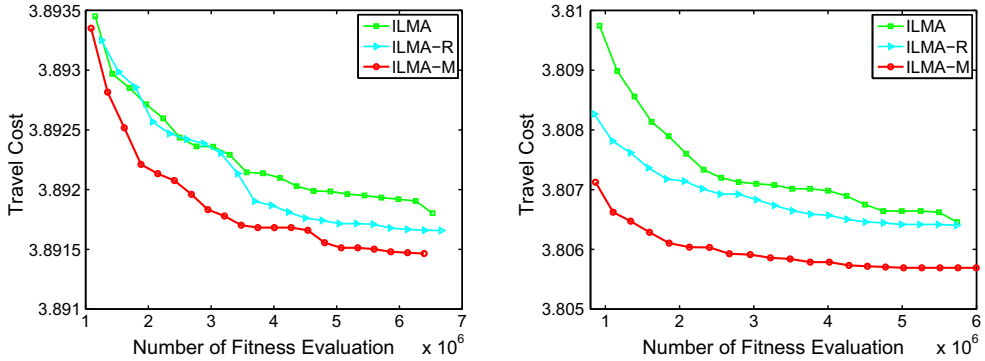
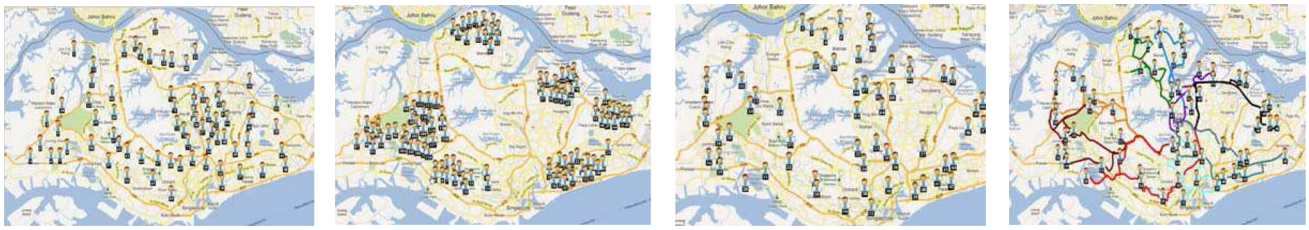


Fig. 9 Averaged search convergence traces (across 30 independent runs) of *ILMA*, *ILMA-R*, and *ILMA-M* on representative CARP "E" and "S" Series instances. *Y*-axis: Travel cost, *X*-axis: Number of Fitness Evaluation. Note that *ILMA-M* is observed to search significantly faster in converging to the near-optimal solution on the respective CARPs than the other counterparts

Table 8 Solution quality of *ILMA*, *ILMA-R*, and *ILMA-M* on *egl* "E" and "S" Series CARP instances

CARP	*ILMA*			*ILMA-R*			*ILMA-M*		(Proposed method)
instance	*B.Cost*	*Ave.Cost*	*Std.Dev*	*B.Cost*	*Ave.Cost*	*Std.Dev*	*B.Cost*	*Ave.Cost*	*Std.Dev*
1.E1A	3548	3548	0	3548	3548	0	3548	3548≈	0
2.E1B	4498	4517.63	12.45	4498	4517.80	13.19	4498	**4513.27**≈	14.93
3.E1C	5595	5599.33	7.56	5595	5601.73	8.84	5595	**5598.07**≈	8.58
4.E2A	5018	5018	0	5018	5018	0	5018	5018≈	0
5.E2B	6317	6341.53	20.15	6317	6344.03	22.38	6317	**6337.90**≈	11.90
6.E2C	8335	8359.87	36.61	8335	8355.07	39.26	8335	**8349.97**≈	26.16
7.E3A	5898	5921.23	30.07	5898	5916.93	30.50	5898	**5910.97**+	30.57
8.E3B	7777	7794.77	23.08	7777	7792.17	29.95	**7775**	**7788.70**≈	15.74
9.E3C	10,292	**10,318.73**	40.89	10,292	10,327.07	33.46	10,292	10,319.16≈	36.15
10.E4A	6461	6471.37	15.16	**6458**	6481.77	22.77	6461	**6469.80**≈	10.27
11.E4B	8995	9060.67	45.29	8993	9067.93	50.54	**8988**	**9053.97**≈	41.49
12.E4C	**11,555**	**11,678.47**	73.57	11,594	11,728.30	82.39	11,576	11,697.27≈	76.98
13.S1A	5018	5023.93	18.14	5018	5025.97	26.97	5018	5023.67≈	25.39
14.S1B	6388	6404.07	22.96	6388	6403.30	20.89	6388	**6392.80**+	14.65
15.S1C	8518	8577.63	44.18	8518	8581.67	33.98	8518	**8576.53**≈	33.12
16.S2A	9920	10,037.43	61.51	9925	10,050.30	54.24	**9896**	**10,010.20**≈	67.13
17.S2B	13,191	13,260.03	45.37	13,173	13,257.90	48.94	**13,147**	**13,245.56**≈	53.02
18.S2C	16,507	**16,605.10**	65.26	16,480	16,626.43	62.90	**16,468**	16,615.40≈	76.79
19.S3A	10,248	10,342.77	47.56	10,278	10,369.40	52.42	**10,239**	**10,339.40**≈	53.29
20.S3B	13,764	13,912.97	79.85	13,779	13,899.70	76.96	**13,749**	**13,881.33**≈	85.78
21.S3C	17,274	17,371.10	79.12	17,277	17,402.43	74.37	**17,261**	**17,355.03**+	48.23
22.S4A	12,335	12,498.47	67.72	12,407	12,534.47	63.23	**12,320**	**12,489.43**≈	83.91
23.S4B	**16,378**	16,542.93	89.65	16,443	16,540.43	87.52	16,415	**16,512.43**≈	57.54
24.S4C	20,613	20,794.80	77.51	20,589	20,841.13	85.53	**20,564**	**20,774.20**≈	86.78
No. Win	2	3		1	0		**10**	**18**	

The superior solution quality of each respective problem instance is highlighted in bold font. "No. Win" denotes the number of instances that an algorithm achieved best performance. Note that *ILMA-M* is superior on 75 % (18/24) of the instances ("≈", "+" and "−" denote *ILMA-M* statistically significant similar, better, and worse than *ILMA*, respectively)



(a) Customer distribution of solved PCP 1

(b) Customer distribution of solved PCP 10

(c) Customer distribution of the new unseen PCP to solve

(d) Optimized routes by *MCES* on the new unseen PCP

Fig. 10 Faster evolutionary optimization of real world PCP by transfer learning from past solving experiences. The head portrait in the subfigures represent the customers to be serviced

can be seen as a variant of the vehicle routing problem with stochastic demand (VRPSD), in which the true demand, i.e., the actual weight and dimensions of the package for each customer remain uncertain before it is serviced on site. In such situations, it may happen that the service vehicle capacity may be insufficient to accommodate the true demand faced (i.e., capacity of the package for example) upon arriving at the customer location, thereby necessitating a return trip to the central depot for capacity replenishment before proceeding to service the affected customers. This not only leads to a substantial increase in the operational costs, but more importantly, probable customer dissatisfactions and unhappiness due to failures in meeting the advertised delivery time guaranteed.

In the present context, the experiences of past optimized PCP routes and the unseen PCP requiring routes design are provided by the courier company. In the PCP of interest here, the archival of past experiences include ten previously solved PCPs of diverse customer distributions and customer size (including uniform, well-clustered, etc., distributed customers, ranging from about 100 to over 500). For the purpose of illustration, two solved PCPs with customer distributions are depicted in Fig. 10a, b. The customer distribution of the new unseen PCP to solve is then depicted in Fig. 10c. On this problem, the recently reported state-of-the-art evolutionary search with Monte Carlo simulation for designing reliable VRPSD solution [70] and labeled here as *CES*, is considered as the baseline evolutionary solver. To speed up the search for reliable PCP routes, the proposed transfer learning culture-inspired operators are incorporated into the *CES*, which is notated here as *MCES*. Thus *MCES* differs from *CES* in that the former is equipped with an initial population of biased routing solutions that are generated using knowledge memes that are learned from search experiences on the ten previously solved PCPs. To ensure a fair comparison, the parameter and operator settings of *CES* and *MCES* are kept consistent to that of [70].

The optimized routes obtained by *MCES* on the new unseen PCP is illustrated in Fig. 10d. With the availability of the knowledge meme mined from past PCP solving experiences, *MCES* attained a population of higher quality solutions than *CES*, right at the start of the search (i.e., initialization). In addition, *MCES* is shown to search more efficiently than the *CES* throughout the entire search. Upon convergence, *MCES* showcased a computational cost savings of more than 58 % to arrive at the same best solution found by counterpart *CES*. It is worth highlighting that as *MCES* and *CES* share a common evolutionary solver, the significant improvements of the former with respect to efficiency can thus be clearly attributed to the proposed transfer learning as culture-inspired operators, which generate high-quality solutions in the initial population to speed up evolutionary search by transfer learning from past experiences.

7 Conclusion

In this paper, we have proposed a new *Memetic Computation Paradigm*: *Evolutionary Optimization* + *Transfer Learning* for search, which models how human solves problems and presented a novel study towards intelligent evolutionary optimization of problems through the transfers of structured knowledge in the form of memes as building blocks learned from previous problem-solving experiences, to enhance future evolutionary searches. In particular, the four culture-inspired operators, namely, *Learning*, *Selection*, *Variation* and *Imitation* have been proposed and presented. The mathematical formulations of the cultural evolutionary operators for solving well established NP-hard routing problems have been derived, where learning is realized by maximizing the statistical dependence between past problem instances solved and the respective optimized solution. In contrast to earlier works, the proposed approach facilitates a novel representation, learning and imitation of generalized knowledge that provide greater scope for faster evolutionary search on unseen related problems. Further, comprehensive studies on the widely studied NP-hard CVRP and CARP domain, has been made to demonstrate and validate the benefits of the proposed faster evolutionary search approach. Subsequently studies on a real world Package Collection/Delivery Prob-

lem further verified the effectiveness of the proposed fast evolutionary approach.

Future works will explore the generality of the proposed paradigm in the context of dynamic optimization problems, where the problems of two subsequent time instances may share high similarity. Last but not the least, although this paper focuses on evolutionary optimization approaches, the proposed cultural-inspired transfer learning operators can also apply to other population based methods, such as particle swarm optimization, differential evolution, etc.

Acknowledgments This work is partially supported under the A*Star-TSRP funding, by the Singapore Institute of Manufacturing Technology-Nanyang Technological University (SIMTech-NTU) Joint Laboratory and Collaborative research Programme on Complex Systems, and the Computational Intelligence Graduate Laboratory at NTU.

References

1. Bransford JD, Brown AL, Cocking RR (2000) How people learn: brain, mind, experience, and school. National Academies Press, Washington
2. Byrnes JP (1996) Cognitive development and learning in instructional contexts. Allyn and Bacon, Boston
3. Reif M, Shafait F, Dengel A (2012) Meta-learning for evolutionary parameter optimization of classifiers. Mach Learn 87(3):357–380
4. Ishibuchi H, Kwon K, Tanaka H (1995) A learning algorithm of fuzzy neural networks with triangular fuzzy weights. Fuzzy Sets Syst 71(3):277–293
5. Tan KC, Chen YJ, Tan KK, Lee TH (2005) Task-oriented developmental learning for humanoid robots. IEEE Trans Ind Electron 52(3):906–914
6. Tan KC, Liu DK, Chen YJ, Wang LF (2005) Intelligent sensor fusion and learning for autonomous robot navigation. Appl Artif Intell 19(5):433–456
7. Nojima Y, Ishibuchi H (2009) Incorporation of user preference into multi-objective genetic fuzzy rule selection for pattern classification problems. Artif Life Robot 14(3):418–421
8. Pan SJ, Yang Q (2010) A survey on transfer learning. IEEE Trans Knowl Des Eng 22(10):1345–1359
9. Taylor ME, Stone P (2009) Transfer learning for reinforcement learning domains: a survey. J Mach Learn Res 10:1633–1685
10. Pelikan M, Hauschild MW, Lanzi PL (2012) Transfer learning, soft distance-based bias, and the hierarchical boa. In: Proceedings of the 12th international conference on parallel problem solving from nature—volume part I, PPSN'12, pp 173–183
11. Bahceci E, Miikkulainen R (2008) Transfer of evolved pattern-based heuristics in games. In: IEEE symposium on computational intelligence and games, 2008. CIG '08, pp 220–227
12. Asta S, Ozcan E, Parkes AJ, Etaner-Uyar AS (2013) Generalizing hyper-heuristics via apprenticeship learning. In: Middendorf M, Blum C (eds) EvoCOP, volume 7832 of Lecture Notes in Computer Science. Springer, Berlin, pp 169–178
13. Iqbal M, Browne W, Zhang MJ (2014) Reusing building blocks of extracted knowledge to solve complex, large-scale boolean problems. IEEE Trans Evol Comput 18:465–580
14. Jin YC (2010) Knowledge incorporation in evolutionary computation. Studies in fuzziness and soft computing. Springer, Berlin
15. Wang H, Kwong S, Jin YC, Wei W, Man K (2005) Agent-based evolutionary approach to interpretable rule-based knowledge extraction. IEEE Trans Syst Man Cybern C 29(2):143–155
16. Tang K, Mei Y, Yao X (2009) Memetic algorithm with extended neighborhood search for capacitated arc routing problems. IEEE Trans Evol Comput 13(5):1159–1166
17. Calian D, Bacardit J (2013) Integrating memetic search into the biohel evolutionary learning system for large-scale datasets. Memetic Comput 5(2):95–130
18. Tang LX, Wang XP (2013) A hybrid multiobjective evolutionary algorithm for multiobjective optimization problems. IEEE Trans Evol Comput 17(1):20–45
19. Tayarani-N MH, Prugel-Bennett A (2013) On the landscape of combinatorial optimization problems. IEEE Trans Evol Comput 18(3):420–434
20. Cheng R, Zhang X, Tian Y, Jin Y (2014) An efficient approach to nondominated sorting for evolutionary multiobjective optimization. IEEE Trans Evol Comput 19:201–213
21. Sutcliffe AG, Wang D (2014) Memetic evolution in the development of proto-language. Memetic Comput 6(1):3–18
22. Ray T, Asafuddoula M, Sarker R (2014) A decomposition-based evolutionary algorithm for many objective optimization. IEEE Trans Evol Comput 19(3):445–460
23. Patvardhan C, Bansal S, Srivastav A (2015) Quantum-inspired evolutionary algorithm for difficult knapsack problems. Memetic Comput 7(2):135–155
24. Ong YS, Keane AJ (2004) Meta-Lamarckian learning in memetic algorithms. IEEE Trans Evol Comput 8(2):99–110
25. Nguyen QC, Ong YS, Lim MH (2009) A probabilistic memetic framework. IEEE Trans Evol Comput 13(3):604–623
26. Gupta A, Ong YS, Feng L (2015) Multifactorial evolution: towards evolutionary multitasking. IEEE Trans Evol Comput (accepted)
27. Neri F, Cotta C, Moscato P (2011) Handbook of memetic algorithms. Studies in computational intelligence. Springer, Berlin
28. Kramer O (2010) Iterated local search with powell's method: a memetic algorithm for continuous global optimization. Memetic Comput 2(1):69–83
29. Tang K, Mei Y, Yao X (2009) Memetic algorithm with extended neighborhood search for capacitated arc routing problems. IEEE Trans Evol Comput 13(5):1151–1166
30. Chen XS, Ong YS, Lim MH, Tan KC (2011) A multi-facet survey on memetic computation. IEEE Trans Evol Comput 5:591–607
31. Dawkins R (1976) The selfish gene. Oxford University Press, Oxford
32. Blackmore S (1999) The meme machine. Oxford University Press, Oxford
33. Ong YS, Lim MH, Chen XS (2010) Research frontier: memetic computation—past, present and future. IEEE Comput Intell Mag 5(2):24–36
34. Chu PC, Beasley JE (1997) A genetic algorithm for the generalised assignment problem. Comput Oper Res 24(1):17–23
35. Jensen MT (2003) Generating robust and flexible job shop schedules using genetic algorithms. IEEE Trans Evol Comput 7(3):275–288
36. Neri F, Toivanen J, Cascella GL et al (2007) An adaptive multimeme algorithm for designing HIV multidrug therapies. IEEE/ACM Trans Comput Biol Bioinform 4(2):264–278
37. Elsayed S, Sarker R, Essam D (2012) Memetic multi-topology particle swarm optimizer for constrained optimization. In: IEEE congress on evolutionary computation
38. Louis SJ, McDonnell J (2004) Learning with case-injected genetic algorithms. IEEE Trans Evol Comput 8(4):316–328
39. Cunningham P, Smyth B (1997) Case-based reasoning in scheduling:reusing solution components. Int J Prod Res 35(4):2947–2961
40. Yang SX, Yao X (2008) Population-based incremental learning with associative memory for dynamic environments. IEEE Trans Evol Comput 12(5):542–561
41. Pelikan M, Hauschild MW (2012) Learn from the past: improving model-directed optimization by transfer learning based on

distance-based bias. Missouri Estimation of Distribution Algorithms Laboratory, University of Missouri in St. Louis, MO, USA. Tech. Rep, 2012007
42. Santana R, Mendiburu A, Lozano JA (2012) Structural transfer using edas: An application to multi-marker tagging snp selection. In: 2012 IEEE congress on evolutionary computation (CEC), pp 1–8
43. Santana R, Armañanzas R, Bielza C, Larrañaga P (2013) Network measures for information extraction in evolutionary algorithms. International Journal of Computational Intelligence Systems 6(6):1163–1188
44. Lynch A (1991) Thought contagion as abstract evolution. J Ideas 2:3–10
45. Brodie R (1996) Virus of the mind: the new science of the meme. Integral Press, Seattle
46. Grant G (1990) Memetic lexicon. In: Principia Cybernetica Web
47. Situngkir H (2004) On selfish memes: culture as complex adaptive system. J Soc Complex 2(1):20–32
48. Heylighen F, Chielens K (2008) Cultural evolution and memetics. In: Meyers B (ed) Encyclopedia of complexity and system science. Springer, Berlin
49. Nguyen QH, Ong YS, Lim MH (2008) Non-genetic transmission of memes by diffusion. In: Proceedings of the 10th annual conference on genetic and evolutionary computation (GECCO '08), (8):1017–1024
50. Meuth R, Lim MH, Ong YS, Wunsch D (2009) A proposition on memes and meta-memes in computing for higher-order learning. Memetic Comput 1:85–100
51. Feng L, Ong Y-S, Lim M-H, Tsang IW (2014) Memetic search with inter-domain learning: a realization between cvrp and car. IEEE Trans Evol Comput. doi:10.1109/TEVC.2014.2362558
52. Minsky M (1986) The society of mind. Simon & Schuster, New York
53. Dantzig G, Ramser JH (1959) The truck dispatching problem. Manag Sci 6:80–91
54. Golden B, Wong R (1981) Capacitated arc routing problems. Networks 11(3):305–315
55. Chen XS, Ong YS, Lim MH, Yeo SP (2011) Cooperating memes for vehicle routing problems. Int J Innov Comput 7(11):1–10
56. Cordeau JF, Laporte G, Mercier A (2001) A unified tabu search heuristic for vehicle routing problems with time windows. J Oper Res Soc 52:928–936
57. Prins C (2004) A simple and effective evolutionary algorithm for the vehicle routing problem. Comput Oper Res 31:1985–2002
58. Reimann M, Doerner K, Hartl RF (2004) D-ants: savings based ants divide and conquer the vehicle routing problem. Comput Oper Res 31:563–591
59. Lin SW, Lee ZJ, Ying KC, Lee CY (2009) Applying hybrid meta-heuristics for capacitated vehicle routing problem. Expert Syst Appl 36(2, Part 1):1505–1512
60. Lacomme P, Prins C, Ramdane-Chérif W (2004) Competitive memetic algorithms for arc routing problem. Ann Oper Res 141(1–4):159–185
61. Mei Y, Tang K, Yao X (2009) Improved memetic algorithm for capacitated arc routing problem. In: IEEE congress on evolutionary computation, pp 1699–1706
62. Feng L, Ong YS, Nguyen QH, Tan AH (2010) Towards probabilistic memetic algorithm: an initial study on capacitated arc routing problem. In: IEEE congress on evolutionary computation, pp 18–23
63. Gretton A, Bousquet O, Smola A, Schölkopf B (2005) Measuring statistical dependence with hilbert-schmidt norms. In: Proceedings algorithmic learning theory, pp 63–77
64. Zhuang J, Tsang I, Hoi SCH (2011) A family of simple non-parametric kernel learning algorithms. J Mach Learn Res (JMLR) 12:1313–1347
65. Song L, Smola A, Gretton A, Borgwardt KM (2007) A dependence maximization view of clustering. In: Proceedings of the 24th international conference on machine learning, pp 815–822
66. Runco MA, Pritzker S (1999) Encyclopedia of creativity. Academic Press, London
67. Chen XS, Ong YS (2012) A conceptual modeling of meme complexes in stochastic search. IEEE Trans Syst Man Cybern C Appl Rev 99:1–8
68. Dijkstra EW (1959) A note on two problems in connection with graphs. Numerische Mathematik 1:269–271
69. Borg I, Groenen PJF (2005) Modern multidimensional scaling: theory and applications. Springer, Berlin
70. Chen X, Feng L, Ong YS (2012) A self-adaptive memeplexes robust search scheme for solving stochastic demands vehicle routing problem. Int J Syst Serv 43(7):1347–1366