A Novel Multi-Task Genetic Programming Approach to Uncertain Capacitated Arc Routing Problem

Mazhar Ansari Ardeh
mazhar.ansariardeh@ecs.vuw.ac.nz
Victoria University of Wellington
Wellington, New Zealand

Yi Mei
yi.mei@ecs.vuw.ac.nz
Victoria University of Wellington
Wellington, New Zealand

Mengjie Zhang
mengjie.zhang@ecs.vuw.ac.nz
Victoria University of Wellington
Wellington, New Zealand

ABSTRACT
Uncertain Capacitated Arc Routing Problem (UCARP) is an NP-hard optimisation problem with many applications in logistics domains. Genetic Programming (GP) is capable of evolving routing policies to handle the uncertain environment of UCARP. There are many different but related UCARP domains in the real-world to be solved (e.g., winter gritting and waste collection for different cities). Instead of training a routing policy for each of them, we can use the multi-task learning paradigm to improve the training effectiveness by sharing the common knowledge among the related UCARP domains. Previous studies showed that GP population for solving UCARP loses diversity during its evolution, which decreases the effectiveness of knowledge sharing. To address this issue, in this work we propose a novel multi-task GP approach that takes the uniqueness of transferable knowledge, as well as its quality, into consideration. Additionally, the transferred knowledge is utilised in a manner that improves the diversity. We investigated the performance of the proposed method with several experimental studies and demonstrated that the designed knowledge transfer mechanism can significantly improve the performance of GP for solving UCARP.

KEYWORDS
Uncertain Capacitated Arc Routing Problem, Genetic Programming, Hyper Heuristics, Multi-task Optimisation

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honoured. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:

1 INTRODUCTION
The Uncertain Capacitated Arc Routing Problem (UCARP) [32] is a combinatorial optimisation problem that simulates a fleet of vehicles with limited capacity which are assigned to serve a collection of tasks in an environment. In this problem, the environment is modelled with an undirected graph $G(V, E)$. The set of nodes $V$ represent the locations of interest in the environment and the set of edges $E$ represent the routes between the locations. In UCARP, the tasks are assumed to be spread over the routes and the vehicles need to traverse the routes to serve the corresponding tasks. The amount of work that is needed to be done for each task is called its demand. Additionally, serving a task and also traversing an edge without a service, incur positive serving and deadheading costs.

UCARP is an extension of the famous Capacitated Arc Routing Problem (CARP) [21, 42]. In CARP, the presence of tasks, task demands, the presence of routes and the deadheading costs are static. However, these static parameters do not reflect the conditions of real-world scenarios and consequently, Mei et al. [32] considered them to be stochastic in UCARP.

Due to the uncertain nature of the problem, most traditional methods for solving CARP do not perform optimally for UCARP [29]. Instead, the approach of equipping vehicles with routing policies allows them to make decisions in real-time based on the most up-to-date information that they perceive from the environment. A routing policy is a mathematical function that takes the environmental state information for an unserved task as input and computes a real-valued number that represents the priority of the task. This allows idle vehicles to calculate the priority of the available tasks and select the most prior task to serve next. The routing policy method has demonstrated great flexibility in uncertain environments.

Manual design of routing policies is an arduous process that demands significant domain expertise. This motivated the utilisation of Genetic Programming as a Hyper-Heuristic (GPHH) and to evolve routing policies automatically (e.g., [13, 29, 45, 47]). In addition to providing an automated process for designing routing policies, Liu et al. [29] demonstrated that their approach outperformed the majority of preceding methods.

In many real-world situations, there are many different but related problems that share many similarities. As a result, instead of solving them separately, it is possible to solve them simultaneously, extract the knowledge common between all problems and improve the GP training process through sharing the common knowledge. In addition to the improved effectiveness that the multi-task approach offers, it mitigates the issue of UCARP problem changes by letting multiple related problems, that pertain to frequent UCARP changes, to be solved in advance and with better effectiveness.

Previous studies [4–6] have indicated that extracting and sharing common knowledge is a challenging task for UCARP. They identified that lack of diversity in the population due to the existence of duplicates and possible early convergence to local optima can have a negative impact on the quality of knowledge transfer during the sequential problem solving process. Although there has never been any effort for proposing a multi-task framework for UCARP, we
expect that the loss of diversity in the population will also negatively affect the effectiveness of knowledge sharing in multi-task frameworks if these issues are not addressed properly.

To address this issue, in this paper we develop an effective multi-task GP algorithm for evolving routing policies for multiple related UCARP domains/instances. The proposed method takes the lack of diversity issue into consideration and demonstrates significant improvement to the GPHH training effectiveness. Specifically, this work is focused on the following research objectives:

- Propose a novel evolutionary multi-task learning algorithm for evolving UCARP routing policies. Our approach utilises a novel mechanism that extracts and shares knowledge based on both its quality and diversity.
- Verify the effectiveness of the proposed algorithm through detailed experimental studies.
- Analyse the components of the proposed knowledge transfer mechanism and investigate their contributions to the overall performance.

This paper is organised as follows. A background review of UCARP, GPHH and evolutionary multi-task learning is given in Section 2. Section 3 presents the proposed algorithm. We investigate the effectiveness of the proposed method in Section 4 with extensive experimental studies. The paper is concluded in Section 5.

2 LITERATURE REVIEW

2.1 UCARP

As an uncertain combinatorial optimisation problem, UCARP is defined on a graph \( G(V, E) \) in which \( V \) and \( E \) are its set of nodes and edges respectively. UCARP models a set of vehicles with a limited capacity \( Q \) that are assigned to serve a collection of tasks that are spread over some of the edges in \( E \). All vehicles are initially stationed at a depot \( v_0 \in V \). The amount of work needed to be served for an edge \( e \in E \) is called its demand, \( d(e) \) and the edge is referred to as a task if \( d(e) > 0 \). The vehicles need to traverse an edge to serve it. Serving a task \( e \in E \) incurs a serving cost \( sc(e) \) and traversing an edge, irrespective of serving any task on it or not, incurs a deadheading cost \( dc(e) \). In contrast to CARP, \( d(e) \) and \( dc(e) \) have a stochastic nature for all \( e \in E \). Furthermore, the presence of tasks and routes are also stochastic. This property resonates better with many real-world situations in which the actual demand values and the actual road/traffic conditions, which affect the deadheading cost, are not static and cannot be predicted beforehand easily.

The solution for a UCARP instance is a set of routes for the vehicles that serves all the tasks in \( G \) with minimum total cost. A solution for UCARP must respect a set of constraints: 1) all vehicle routes must start from and end with the depot node \( v_0 \); 2) vehicles cannot serve if their capacity is exhausted; 3) a task must be served exactly once [32, 39]. If the capacity of a vehicle is exhausted, it needs to return to the depot to replenish it.

Due to the uncertain nature of UCARP, vehicles may encounter unexpected situations that can disrupt their course. Arriving at a task, a vehicle may find the actual demand to be greater than its remaining capacity. In this case, referred to as route failure, the vehicle has to return to depot, replenish its capacity and then, resume serving the task. Also, a vehicle may need to traverse an edge \( e \) but when arrived at it, it may discover the road to be blocked, i.e. \( dc(e) = \infty \). In this case, called edge failure, the vehicle needs to find an alternative way around it.

UCARP is an NP-hard problem [32] and as a result, it is not possible to devise exact methods for solving it. Evolutionary algorithms, on the other hand, have demonstrated very good performance in this regard. In general, the proactive and reactive methods are the two categories of evolutionary approaches for solving UCARP.

Proactive methods typically utilise a meta-heuristic algorithm for finding robust solutions and optimising them for the environmental uncertainties. These methods do not take the uncertain nature of UCARP directly into consideration but mostly rely on optimising the robustness of solutions. Methods in [32, 38] are good examples of the proactive approach.

Reactive methods form the category of approaches that rely on the concept of routing policies. A routing policy is a real-valued mathematical function that can calculate the priority of an unserved task based on the most recent information that the vehicle has about its environment. This allows the vehicle to select the next task to serve in real-time and based on their most up-to-date priorities. As a result, these methods exhibit great flexibility for handling the uncertainties of UCARP. Although possible [3], routing policies are not predominantly designed manually as it can be a time-consuming, expensive and problem-specific process. On the other hand, the Genetic Programming algorithm can be utilised as a Hyper-Heuristic for evolving the routing policies more efficiently.

2.2 GPHH

Inspired by the famous Genetic Algorithm, Genetic Programming (GP) is an evolutionary method that can evolve mathematical abstractions of computer programs [1, 2, 15, 25]. GP relies on the crossover, mutation and reproduction genetic operators for evolving a population of programs iteratively. The specifics of these operators depend on the way that the programs are represented, i.e. tree representation. After adopting a representation, GP initialises a random population of computer programs and until some stopping criteria is met, it evaluates the fitness of population members and generates a new population from the current one with the genetic operators and based on the fitness of its members. When the computer programs are heuristics, the GP approach for evolving them is referred to as Genetic Programming Hyper-Heuristics (GPHH).

UCARP routing policies are heuristics as they construct the final solution, i.e. set of vehicle routes. Therefore, the GPHH algorithm can evolve them automatically and Liu et al. [29] were the first to propose using GP as a Hyper-Heuristic for evolving routing policies that are represented as trees and by doing so, they founded the category of reactive methods for solving UCARP. In their approach, the routing policies are not the final routing solutions for UCARP but they are utilised to generate the solution in real-time.

A routing policy assesses the priority of unserved tasks and equipped with such a policy, vehicles can generate the solution in real-time. More specifically, initially all vehicles are idle at the depot. When a vehicle is idle, it considers all the unserved tasks and assesses the priority of each one with its routing policy. Then, it selects the most prior task to serve next and proceeds with serving it without considering any other task. If a routing or an edge failure occurs, the vehicle needs to handle it accordingly until the task
is finished. At this point, the vehicle becomes idle and can select a new task to serve next. If all the tasks are served, then vehicles return to the depot to finish generating the solution [29].

Liu et al. [29] initially proposed their method for a single vehicle. Later on, Mei et al. [33] extended the work for the case of multiple vehicles and Maclachlan et al. [31] proposed a method for enabling the vehicles to collaborate. Liu et al. [30] are the first to propose a hybrid proactive-reactive approach in which the proactive component of their algorithm is comprised of an estimation of distribution algorithm [36] that evolves a route sequence and the reactive component evolves a recourse to guide the vehicle to the depot whenever a route failure occurs. Noting that GP-evolved UCARP routing policies tend to be large and difficult to interpret, Wang et al. [40] proposed evolving an ensemble of smaller but more interpretable polices. Ansari Ardeh et al. [7, 8] were the first to address the scenario change issue of UCARP by proposing transfer learning methods for solving the new problem more effectively.

2.3 Evolutionary Multi-Task Optimisation

Evolutionary multi-task (EMT) optimisation is a search paradigm that pertains to the concurrent optimisation of multiple problems that have equal importance [35]. In this paradigm, as the optimisation process proceeds, the common knowledge, obtained from solving each task, is updated and shared between all tasks. The act of knowledge sharing allows tasks to be solved more effectively and efficiently [23, 46]. Evolutionary multi-task optimisation has been applied successfully to various problems from different fields such as combinatorial optimisations [43, 48], symbolic regression [50], image classification [11] and continuous optimisation [14, 22, 26].

Generally, the EMT approaches fall into two categories of explicit and implicit methods. The implicit methods utilise one single population to solve multiple tasks. In this direction, each individual has a skill factor that determines which task it solves. Solving multiple tasks with a single population facilitates knowledge sharing with the genetic operators [22]. Gupta et al. [22] proposed one of the earliest implicit EMT methods, called Multifactorial Evolutionary Algorithm, MFEA, in which knowledge transfer is achieved through assortative mating, that is performing crossover on individuals with different skill factors. While [22], assortative mating is regulated with a parameter, Da et al. [9] proposed an online approach for adjusting this parameter and hence, the amount of knowledge transfer between tasks. Da et al. also proved theorems that specify the conditions on which EMT can converge. Zheng et al. [49] proposed an extension to MFEA by providing a way of measuring the similarity of tasks and allowing knowledge transfer based on it. Feng et al. [18] extended and MFEA and applied it for solving the generalised vehicle routing problem with occasional drivers.

In the explicit approach, as the name suggests, each task is solved explicitly by a separate and independent population and the exchange of knowledge is achieved through some form of information immigration. Lin et al. [28] devised an explicit EMT for solving multi-objective continuous problems. In their method, a Naive Bayes classifier learned incrementally to detect if an immigrant is likely to improve the target population or not. Lin et al. [27] speculated that if an immigrant was a successful transfer in the previous generation, then its neighbours in the current population, that it originally belonged to, are also more likely to be a successful transfer. In their method, designated as EMT/ET, the immigrants may go through a mutation with some probability to introduce some diversity. Zhang et al. [44] proposed a method in which separate populations solve different job-shop scheduling problems and the crossover operator is modified to mate parents from different populations to transfer knowledge. Feng et al. [19] proposed an explicit EMT method, denoted as EMEA, that utilises autoencoders for adapting immigrants to the search space of the target population for solving continuous optimisation problems. Feng et al. [20] extended EMEA for solving dynamic problems. In a later work, Feng et al. [17] extended the autoencoder-based adaptation method for discrete combinatorial problems.

Despite the interest that evolutionary multi-task optimisation is receiving from scholars, it is still in an early stage. Specifically, the number of multi-task works for GP in general and, GPHH in particular, is very limited. This makes the number of available algorithms for performing multi-task learning on UCARP very few. The issue of possible early convergence to local optima and lack of diversity, that have been shown to negatively impact knowledge transfer for UCARP [4–6], can make the situation even more challenging. As a result, in this work, we propose a novel multi-task learning method that addresses the lack of diversity issue and investigate it in depth.

3 DIVERSITY-DRIVEN MULTI-TASK GPHH

The quality of multi-task learning depends greatly on the quality of the knowledge that is exchanged between the tasks. Accordingly, if the transfer of knowledge does not convey any information that is not novel for the target task then naturally, there can be no benefit in multi-task optimisation. This is particularly important for a problem such as UCARP that is susceptible to losing its diversity during the GP evolution. What makes the issue even more challenging is the fact that genotypically different GP trees can demonstrate the same phenotypic behaviour [24]. Consequently, for a successful exchange of knowledge, the possibility that the target task may contain phenotypic duplicates of the knowledge being transferred must be considered and handled.

Accordingly, Algorithm 1 presents our Diversity-Driven Multi-Task GP, DDMTGP, approach for solving UCARP. The algorithm receives $k$ different but related UCARP instances and returns $k$ best solutions for each instance. The algorithm has two parameters: the number of immigrants $\eta$ that populations send to transfer knowledge, and the number of trials $\theta$ to find a novel immigrant. DDMTGP starts the search process by creating $k$ initial populations randomly, one for each UCARP instance. Then, for MaxGen number of generations, the algorithm evolves these populations. For this, first each population is evaluated, the best routing policy is recorded, and then $\eta$ number of immigrants are selected with tournament selection based on the fitness value. After this, each population breeds a new population with the standard crossover, mutation and reproduction operators [25]. After this phase, the process of transferring knowledge through exchange of immigrants begins.

In DDMTGP, knowledge exchange is focused on transferring novel information that the receiving population does not possess. Hence, during the transfer of immigrants from population $i$ to population $j$, population $j$ decides which of its members should be
displaced in favour of the incoming individuals. This decision is made with the SelectToReplace algorithm that searches the population for redundant phenotypic duplicates that can be replaced with the incoming information. To improve the efficiency of this process, the unique individuals in each population are encoded into a hash table with the Hash method described in Subsection 3.2. The process of identifying the phenotypically duplicate routing policies is described in Subsection 3.1 and, the algorithm for selecting the individuals to be replaced is given in Subsection 3.3.

Insertion of immigrants into the population is also permitted if the target population does not contain them. Accordingly, for each immigrant, DDMTG uses the hashed summary of the population to check if the target population contains the phenotypic information of the immigrant and if not, the immigrant is inserted into the population. Otherwise, DDMTG utilises the standard mutation operator for \( q \) times to find a new individual in the vicinity of the immigrant. Finally, the best individuals are returned as the outcome.

### 3.1 Phenotypic Characterisation

In the context of UCARP, while the genotypic characteristics of a routing policy pertain to the genetic materials in the tree structure, its phenotypic characteristics refer to the tasks that vehicles select to serve next, based on the state of the vehicles and their environment. More specifically, when a vehicle is in a situation in which it needs to make a decision on what task to serve next, it utilises the routing policy to rank the unserved tasks to select the task with the highest priority to serve next. Therefore, if two routing policies calculate the highest priority for the same task in the same situation, then they are phenotypically similar in that decision-making situation.

Based on this observation, the phenotypic behaviour of a routing policy \( \varphi \) can be characterised with a set of decision-making situations \( \Omega \). For this, first each task in each situation is indexed and the index is kept fixed. Then, \( \varphi \) is utilised to rank each task in the decision situation \( \Omega_i \) of \( \Omega \). Finally, \( \xi(\varphi) \), the phenotypic characterisation of \( \varphi \) in the form a numerical vector, is obtained by retrieving the index of the most prior task in each situation.

Having the concept of characterisation vector developed, it is possible to identify the phenotypically unique routing policies. Accordingly, two routing policies \( \varphi_1 \) and \( \varphi_2 \) are considered similar if they have the same characterisation vector, i.e. \( \xi(\varphi_1) = \xi(\varphi_2) \).

Our approach to measuring the phenotypic similarity is particularly different from the method by Hildebrandt et al. [24] for phenotypic characterisation of scheduling rules. In contrast to the work in [24], we do not need the characterisation vector to calculate the similarity of policies and hence, we do not need the overhead of calculating the phenotypic characterisation for the reference rule.

### 3.2 Locating Unique Individuals

To detect duplicates in a population, DDMTG encodes the population into a hash table that allows the quick and efficient query of its content. For this purpose, first it is needed to hash a routing policy into an integral value. Having the ability to characterise a routing policy into an integral phenotypic vector, as described in Subsection 3.1, any hashing mechanism that provides a robust guarantee against collision, i.e. assign the same hash value to different vectors, can be utilised. In this work, the left-shifting approach in Algorithm 2 demonstrated this property. Given \( \xi_\varphi \), the characterisation vector of a routing policy \( \varphi \), as input, the LHash method computes its
Algorithm 3: SelectToReplace(Popi, Φi, η)

Input: A population Popi, to select some of its members to be discarded; Φi, a hash table summary of Popi; η number of members to be discarded.
Output: A set of individuals to be discarded from Popi.

1. \( \Psi \leftarrow \emptyset \) // Set of duplicates in Popi.
2. For \( \varphi \in \text{Popi} \), do

3. \( \zeta_{\varphi} \leftarrow \zeta(\varphi) \)
4. \( h_{\varphi} \leftarrow \text{Hash}(\zeta_{\varphi}) \)
5. If \( h_{\varphi} \notin \Phi_i \) then
6. \( \Psi \leftarrow \Psi \cup \{\varphi\} \)
7. end
8. end
9. If |\( \Psi \) | < \( \eta \) then
10. \( \Psi \leftarrow \Psi \cup \{\text{TournamentSelectWorst}(\text{Popi}, \eta - |\Psi|)\} \)
11. end
12. return \( \Psi \)

hash value based on each component of \( \zeta_{\varphi} \) with the formula in line 5 of the algorithm in which \( << \) represents the left-shift bit operator.

Having the LSHash method, it is fairly straightforward to implement the Hash method used in Algorithm 1 for summarising unique individuals in a population Popi. Our implementation is presented in Algorithm 2 in which first an empty hash \( \Phi_i \) table is initialised. Then, for each routing policy \( \varphi \in \text{Popi} \), its hash value is calculated with LSHash, the hash value is checked against \( \Phi_i \) and if the table does not contain it, then the \( \varphi \) is inserted into the table.

3.3 Selecting Replaceable Individuals

In DDMTGp, each population needs to select one of its members to be discarded and replaced with each incoming immigrant. In our approach, the focus is on finding the individuals with the lack of phenotypic diversity and hence, in DDMTGp, the duplicates in the population are candidates for being discarded first. Accordingly, Algorithm 3 presents our proposed method.

This method receives a population of routing policies as input and returns the phenotypic duplicates in the population as output. The algorithm achieves this with a hash table data structure \( T \) that keeps track of unique individuals in the population. For this, the algorithm first calculates the phenotypic characterisation \( \zeta_{\varphi} \) for each routing policy \( \varphi \) according to the description in Subsection 3.1. Then, it hashes the vector \( \zeta_{\varphi} \) into a unique integer value and looks for the hash value in a hash table data structure \( T \). If \( T \) does not contain the hash value, then the routing policy is unique and hence, it is added to \( T \). Otherwise, the policy is a duplicate and is recorded to be replaced by immigrants. The main reason for utilising a hash table is that its \( O(1) \) time complexity for insertion and search eliminates the need for iterating the population with a second loop to check if any other phenotypically identical policy exists. Finally, if the number of duplicates is less than needed, the remaining individuals are selected from the worst members of the population with a tournament selection.

4 EXPERIMENTAL STUDIES

To investigate the effectiveness of DDMTGp, we conducted experiments on a wide range of UCARP instances based on the gdb dataset [32]. For this, we utilised DDMTGp to run several multi-task scenarios in which three related problems were solved simultaneously. Mei et al. [33] showed that a change in the number of vehicles, even as small as one, can create a significantly different problem. Hence, in all scenarios, the difference between problems is based only on the number of vehicles. Table 2 presents these scenarios. For DDMTGp, we experimented with a wide range of values for the \( \eta \) and \( \delta \) parameters and obtained the best result with \( \eta = 200 \) and \( \delta = 10 \). Table 1 presents the GP terminals for describing a UCARP environment, GP functions and parameters. The / division operator is protected and will return 1 if its denominator is zero. We used 5 instances for training and 500 for testing the policies [29]. All experiments were conducted for 30 independent runs.

4.1 Performance Analysis

The literature of multi-task optimisation algorithms for GPHH is very limited and most of the available GP-based approaches are not adapted to handle the population diversity problem of UCARP. Hence, we selected the EMETET method [27] because, similar to our approach, it also relies on the mutation operator for introducing diversity into the transferred knowledge. Additionally, the authors compared EMETET with several existing approaches, including MFEA [22] and EMEA [9], and demonstrated its superiority and therefore, it can be a good representative for the existing methods. EMETET was originally developed for continuous multi-objective optimisation problems and we adapted it for UCARP by using the phenotypic distance measure proposed in [24]. Additionally, we also considered an Island-based Multi-Task approach, IMT [10, 41], that selects individuals for transfer and replacement based on their fitness with tournament selection.

Table 3 presents the mean test performance of the compared algorithms, in which the best results are highlighted in boldface. As is evident, DDMTGp has achieved better performances on almost all experiments. To rank the results and verify the existence of significant difference, we applied the Friedman test with a \( \alpha = 0.05 \) confidence level on the results. The ranks and the obtained \( p \)-value
are given in the bottom rows of Table 3. The test ranks DDMTG as the best and the p-value indicates that there is a significant difference. In order to find the difference, we conducted the Nemenyi post-hoc test [34] and the p-values for the pairwise comparison of the algorithms is given in Table 4. As is evident in Table 4, the performance of DDMTG is significantly different from the compared algorithms. This observation, alongside the algorithms rank, demonstrate the superiority of DDMTG.

Figure 1 presents the convergence curve of the compared algorithms for some of the experiments. As is evident, all methods started from a similar state but as the evolution proceeded, DDMTG was able to reach to better states and maintain it later on. Figure 2 presents the distribution of the results achieved by different algorithms for some of the experiments in the form of violin plots. The plots demonstrate that the results obtained by DDMTG have better median and are distributed towards lower fitness values and confirm the superiority of DDMTG to other methods. We observed similar patterns in other experiments.

4.2 Phenotypic Diversity of GP Population

As it was mentioned in Section 1, the GP approach for solving UCARP tends to suffer from losing population diversity which can negatively impact the act of knowledge sharing. This observation was the fundamental motivation for our DDMTG. Accordingly, in this section we investigate the impact that our modifications to the GP mechanism has had on the GP population diversity. For this matter, we utilised the entropy measure [12] to quantify the population phenotypic diversity. Inspired by thermochemical systems, entropy measures the amount of population disorder. Accordingly, an increase in entropy represents an increase in diversity and it has been shown that population may be stuck in local optima if entropy does not change or decreases monotonically [12]. To calculate the entropy, we utilised the DBScan clustering algorithm [16] with the phenotypic distance measure in [24] and a cluster radius of zero to group similar individuals into the set of clusters C and calculated the entropy of the population Pop as $ entropy(Pop) = -\sum_{c \in C} \frac{1}{|c|} \log(\frac{|c|}{|Pop|}) $.

Table 5 presents the entropy of the final population, averaged over 30 runs. It is obvious from the table that DDMTG has better final population entropy and hence, diversity. Additionally, Figure 3 depicts the population entropy over GP generations for a few experiments. As can be seen, since the populations are initialised randomly, all algorithms start with a high level of diversity. However, as the evolution proceeds, all existing algorithms lose

![Figure 1: Convergence curve of the compared methods.](image)

### Table 2: Multi-task scenarios used in the experiments

<table>
<thead>
<tr>
<th>MT Scenario</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prb Dataset</td>
<td>gdb1</td>
<td>gdb1</td>
<td>gdb1</td>
<td>gdb1</td>
<td>gdb1</td>
<td>gdb1</td>
<td>gdb1</td>
</tr>
<tr>
<td>No. Vehicles</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
</tbody>
</table>

![Image](image)

### Table 3: The test performance of the compared algorithms

<table>
<thead>
<tr>
<th>Prb</th>
<th>GPHH</th>
<th>EMTET</th>
<th>IMT</th>
<th>DDMTG</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>354.0 ± 3.6</td>
<td>355.7 ± 3.3</td>
<td>356.0 ± 3.4</td>
<td>354.3 ± 2.7</td>
</tr>
<tr>
<td>2</td>
<td>347.2 ± 5.1</td>
<td>344.5 ± 4.0</td>
<td>343.7 ± 3.9</td>
<td>338.9 ± 2.1</td>
</tr>
<tr>
<td>3</td>
<td>360.3 ± 3.1</td>
<td>354.5 ± 3.7</td>
<td>355.3 ± 4.3</td>
<td>350.5 ± 1.8</td>
</tr>
<tr>
<td>4</td>
<td>383.5 ± 4.2</td>
<td>383.0 ± 1.8</td>
<td>383.4 ± 1.5</td>
<td>381.5 ± 2.1</td>
</tr>
<tr>
<td>5</td>
<td>367.6 ± 4.9</td>
<td>364.3 ± 3.1</td>
<td>364.8 ± 3.4</td>
<td>360.5 ± 2.3</td>
</tr>
<tr>
<td>6</td>
<td>382.8 ± 3.3</td>
<td>382.5 ± 1.9</td>
<td>382.6 ± 1.9</td>
<td>379.0 ± 3.7</td>
</tr>
<tr>
<td>7</td>
<td>309.3 ± 6.6</td>
<td>310.1 ± 5.6</td>
<td>310.4 ± 7.8</td>
<td>301.9 ± 3.7</td>
</tr>
<tr>
<td>8</td>
<td>305.6 ± 1.2</td>
<td>306.0 ± 0.4</td>
<td>305.7 ± 0.9</td>
<td>304.4 ± 2.1</td>
</tr>
<tr>
<td>9</td>
<td>326.0 ± 4.7</td>
<td>326.4 ± 3.0</td>
<td>324.8 ± 4.2</td>
<td>320.7 ± 5.3</td>
</tr>
<tr>
<td>10</td>
<td>340.8 ± 4.4</td>
<td>341.4 ± 3.7</td>
<td>340.3 ± 3.0</td>
<td>336.8 ± 4.6</td>
</tr>
<tr>
<td>11</td>
<td>318.0 ± 0.5</td>
<td>318.6 ± 0.6</td>
<td>318.4 ± 0.7</td>
<td>318.3 ± 0.6</td>
</tr>
<tr>
<td>12</td>
<td>321.4 ± 5.2</td>
<td>321.6 ± 3.8</td>
<td>321.4 ± 3.5</td>
<td>318.6 ± 1.2</td>
</tr>
<tr>
<td>13</td>
<td>347.6 ± 2.5</td>
<td>346.9 ± 3.0</td>
<td>346.9 ± 3.0</td>
<td>345.6 ± 3.6</td>
</tr>
<tr>
<td>14</td>
<td>420.7 ± 5.4</td>
<td>417.7 ± 4.9</td>
<td>420.1 ± 4.9</td>
<td>415.8 ± 4.0</td>
</tr>
<tr>
<td>15</td>
<td>457.9 ± 5.8</td>
<td>457.7 ± 4.1</td>
<td>459.6 ± 4.4</td>
<td>434.3 ± 2.5</td>
</tr>
<tr>
<td>16</td>
<td>347.2 ± 6.1</td>
<td>346.3 ± 4.8</td>
<td>347.2 ± 5.8</td>
<td>342.8 ± 4.3</td>
</tr>
<tr>
<td>17</td>
<td>340.5 ± 4.7</td>
<td>337.2 ± 3.5</td>
<td>337.4 ± 3.1</td>
<td>333.8 ± 3.4</td>
</tr>
<tr>
<td>18</td>
<td>351.5 ± 2.5</td>
<td>351.4 ± 0.4</td>
<td>350.8 ± 2.1</td>
<td>349.5 ± 3.4</td>
</tr>
<tr>
<td>19</td>
<td>376.1 ± 7.6</td>
<td>378.3 ± 7.3</td>
<td>375.9 ± 8.2</td>
<td>368.6 ± 4.7</td>
</tr>
<tr>
<td>20</td>
<td>349.7 ± 1.5</td>
<td>349.3 ± 1.1</td>
<td>348.9 ± 1.5</td>
<td>347.4 ± 1.8</td>
</tr>
<tr>
<td>21</td>
<td>356.6 ± 1.6</td>
<td>355.8 ± 0.9</td>
<td>356.1 ± 1.3</td>
<td>355.6 ± 0.1</td>
</tr>
<tr>
<td>Rank</td>
<td>3.33</td>
<td>2.74</td>
<td>2.83</td>
<td>1.1</td>
</tr>
<tr>
<td>Friedman’s p-value</td>
<td>5.88e-11</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 4: The p-values for the post-hoc pairwise comparisons of the compared algorithms

<table>
<thead>
<tr>
<th></th>
<th>GPHH</th>
<th>EMTET</th>
<th>DDMTG</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMT</td>
<td>1</td>
<td>0.811</td>
<td>0</td>
</tr>
<tr>
<td>EMTET</td>
<td>−</td>
<td>8e-05</td>
<td>0.00022</td>
</tr>
</tbody>
</table>

![Image](image)
Figure 2: The distribution of the results achieved by DDMTG and some existing methods.

As is evident, the diversity component of DDMTG plays an important role in its performance. This raises two questions: 1) is the better performance caused by just increasing the diversity (i.e. is the contribution of knowledge transfer insignificant)? 2) which component of the diversity mechanism contributes most to the performance? The diversity component has two components itself: 1) sending unique individuals from a source to a target task and, 2) replacing duplicates in the target task with the transferred knowledge. To answer the first question, we equip the single-task

Table 5: Average entropy of the final GP population

<table>
<thead>
<tr>
<th>Prb</th>
<th>GPHH</th>
<th>EMTET</th>
<th>IMT</th>
<th>DDMTG</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.97</td>
<td>3.49</td>
<td>2.68</td>
<td>6.30</td>
</tr>
<tr>
<td>2</td>
<td>3.20</td>
<td>3.76</td>
<td>2.72</td>
<td>6.28</td>
</tr>
<tr>
<td>3</td>
<td>2.78</td>
<td>3.54</td>
<td>2.75</td>
<td>6.27</td>
</tr>
<tr>
<td>4</td>
<td>2.92</td>
<td>3.57</td>
<td>3.02</td>
<td>6.23</td>
</tr>
<tr>
<td>5</td>
<td>3.06</td>
<td>3.55</td>
<td>2.91</td>
<td>6.29</td>
</tr>
<tr>
<td>6</td>
<td>2.50</td>
<td>3.21</td>
<td>2.70</td>
<td>6.23</td>
</tr>
<tr>
<td>7</td>
<td>3.17</td>
<td>3.40</td>
<td>2.58</td>
<td>5.55</td>
</tr>
<tr>
<td>8</td>
<td>3.35</td>
<td>3.79</td>
<td>2.46</td>
<td>5.51</td>
</tr>
<tr>
<td>9</td>
<td>3.11</td>
<td>3.16</td>
<td>2.54</td>
<td>5.89</td>
</tr>
<tr>
<td>10</td>
<td>2.88</td>
<td>3.34</td>
<td>2.93</td>
<td>6.29</td>
</tr>
<tr>
<td>11</td>
<td>3.22</td>
<td>3.48</td>
<td>2.99</td>
<td>6.29</td>
</tr>
<tr>
<td>12</td>
<td>3.23</td>
<td>3.51</td>
<td>3.04</td>
<td>6.16</td>
</tr>
<tr>
<td>13</td>
<td>2.29</td>
<td>3.31</td>
<td>2.65</td>
<td>6.12</td>
</tr>
<tr>
<td>14</td>
<td>3.16</td>
<td>3.36</td>
<td>2.79</td>
<td>6.29</td>
</tr>
<tr>
<td>15</td>
<td>2.64</td>
<td>3.39</td>
<td>2.74</td>
<td>6.24</td>
</tr>
<tr>
<td>16</td>
<td>3.02</td>
<td>3.93</td>
<td>2.79</td>
<td>6.29</td>
</tr>
<tr>
<td>17</td>
<td>3.03</td>
<td>3.55</td>
<td>2.88</td>
<td>6.29</td>
</tr>
<tr>
<td>18</td>
<td>2.70</td>
<td>3.47</td>
<td>2.62</td>
<td>6.20</td>
</tr>
<tr>
<td>19</td>
<td>2.92</td>
<td>3.76</td>
<td>2.98</td>
<td>6.31</td>
</tr>
<tr>
<td>20</td>
<td>2.56</td>
<td>3.53</td>
<td>2.93</td>
<td>6.34</td>
</tr>
<tr>
<td>21</td>
<td>2.68</td>
<td>3.48</td>
<td>2.99</td>
<td>6.23</td>
</tr>
</tbody>
</table>

Figure 3: The distribution of the results achieved by DDMTG and some existing methods.

Table 6-7 present the performance and pairwise comparisons of each method. According to these tables, DDMTG is significantly better than ClGPHH, which indicates the increased diversity is not the only contributing factor to the performance of DDMTG, thus answering our first question. To answer the second question, we note that DDMTG-TS is ranked better than ClGPHH and, although DDMTG-TR is ranked slightly better than GPHH, their difference is not significant while DDMTG-TS is significantly better than GPHH. This shows the algorithm component that replaces the duplicates with the transferred knowledge has a significant contribution to the performance of DDMTG. Nevertheless, the best performance is achieved when both components are combined.

4.3 Algorithm Components

The proposed DDMTG has two main components: 1) checking if the target population contains the transferable individuals and, 2) mutating them when they are seen in the target population. To investigate the effect of each part by disabling each component and measuring its impact on the performance of DDMTG. When the first component is disabled, the algorithm does not check the target population and will send the individuals anyway, which also eliminates the need for mutation. However, to simulate the effect of the mutation, we modify the algorithm to transfer individuals between populations by selecting \( \eta \) individuals from a source population, creating \( \eta - 1 \) new individuals from each selected individual by performing mutation on it and transferring the set of selected individuals and their mutations (\( \eta \) individuals). We refer to the resulting algorithms as DDMTG Without Diversity checking, DDMTG-WD. To disable the second component, we set the \( \theta = 1 \)
in DDMTG (setting $\theta$ to 0 will convert DDMTG to IMT) and refer to it as DDMTG Without Mutation, DDMTG-WM.

Table 8 presents the average performance of each algorithm, their rank and the $p$-value of the Friedman test. The post-hoc pairwise comparisons of the algorithms are given in Table 9. The Friedman test ranks DDMTG as the best amongst the algorithms. However, the post-hoc comparisons reveal that the difference between DDMTG and DDMTG-WM is not significant. On the other hand, DDMTG-WD, which does not check for diversity, is significantly worse than DDMTG, while DDMTG-WM and DDMTG are statistically similar.

5 CONCLUSIONS

The goal of this paper is to develop a novel multi-task learning algorithm for GP to evolve effective routing policies for UCARP. The goal has been successfully achieved by proposing a multitask GP algorithm which considers the possibility that the transferable knowledge may be redundant in the target task. Specifically, we devised a method for discovering and handling this situation. Additionally, we also considered the transfer of knowledge as the opportunity for improving the population diversity.

Our experimental results show that the proposed method can improve the effectiveness of GP and achieve significantly better performance than the state-of-the-art GP approaches on almost all the tested datasets. The experiments also verified the effectiveness of the two major contributions of DDMTG, i.e. the duplicate removal and knowledge transfer mechanism. We have shown that the duplicate removal can greatly increase the diversity of the population and reduces the chance of getting trapped into poor local optima. Furthermore, the new knowledge transfer mechanism can share complementary knowledge among the problems, leading to much better results than simply increasing population diversity.

For our future work, we consider the case in which the transferable knowledge may have been seen in earlier GP generations, which can also make the knowledge redundant and even harmful. We aim to address this in later works. Furthermore, we will also consider the possibility that the source task can also contain duplicates, which increases the chance of transferring redundant knowledge and will investigate this case in future.
REFERENCES


