International Journal of Research in Computing (2820-2139)
Volume 02, No. 01, July 2023.

AGH; An Ant Genetic Hybrid Solution to Solve the Multi-model Traveling Salesmen Problem

H Hansani and MKA Ariyaratne
Department of Computer Science, Faculty of Applied Sciences, University of Sri Jayewardenepura, Gangodawila, Nugegoda, Sri Lanka

Corresponding author: MKA Ariyaratne, Email: mkanuradha@sjp.ac.lk

ABSTRACT The concept of multi-model optimization brings the idea of finding all or most of the existing high quality solutions. Recent research on multi-model optimization (MMO) seemed to be using nature inspired algorithms in solving such interesting problems. Multi-model traveling salesman problem is an important but rarely addressed discrete MMO problem. This paper proposes a hybrid algorithm combining the Ant Colony Systems algorithm (ACS) with a modified genetic algorithm (MODGA) to solve multi-model traveling salesman problems (MMTSPs). The concept of the hybrid algorithm divides the solution into two parts where ACS is used to find an average quality solution which is then provided as a threshold to the MODGA to find other quality solutions as much as possible. Benchmark multi-model TSP problems have been used on the new algorithm to test its capability. 70% of the success PR and 0.6% of success SR values indicates the capability of the method solving MMTSPs. The results compared with several state of the art multi-model optimization algorithms showed that the proposed hybrid algorithm performs competitively with these algorithms. As the first approach to solve MMTSPs without niching strategies, improvements will lead the current algorithm to a greater place.

KEYWORDS: Multi-model Optimization, GA, ACS, Traveling salesmen problem

I Introduction

There are many situations in the real world where we need to come up with multiple different good solutions than one. Particular advantages are when various practical constraints involve with the solutions. In applied mathematics, such problems are called multi-model optimization problems (MMOP) [1]. Multi-model optimization discusses finding several optimum solutions in a single run of an algorithm. When it comes to solving such MMO problems there are many approaches discussed in the literature. One way of achieving these optimal solutions is to run a particular algorithm many times by approaching different areas in the search space of the problem. In many of these cases there is no guarantee to find all or most of all the solutions by such a method. When the problem is in the discrete domain, the complexity of achieving multiple solutions is doubled.

In the research literature on solving such multi-model optimization problems, continuous domain is highly touched and the discrete problems are less addressed [2]–[5]. To date, traveling salesman problem (TSP) is one of the most popular and intensively studied problems in optimization [6]–[8]. In many practical applications related to TSP, several better solutions are required than obtaining a single best solution. For example, it is always recommended for route planning applications to provide several acceptable routes so that the driver can choose the cheapest route based on his own knowledge. Since the road condition changes dynamically, a route can become invalid due to traffic jam or road maintenance. In these cases, drivers want to quickly switch to a candidate route of the same quality in order to get their job done on time. This is where the importance of multimodal TSP (MMTSP) comes to the stage. Among different approaches used to solve both continuous and discrete multi-model optimization problems, use of meta-heuristics such as evolutionary computing or swarm intelligence algorithms are undoubtedly popular due to their population behavior [9]–[12]. In this kind of research work where MMOPs were addressed using meta-heuristics, use of niching strategies is popular [2], [5], and [13]–[15]. Niching brings the idea of dividing the population of solutions into disjoint sets, with the intension of having at least one member in each region of the objective function. Drawbacks of such niching methods can be pointed out as problems with maintaining found solutions, difficulties in scalability and performance measuring, and problems arise with niching parameters. Hence the aim of this paper is twofold; to address discrete multi-model optimization problems; particularly MMTSP, and to use meta-heuristics to solve MMTSP without using niching methods.

Population based stochastic meta-heuristics such as Evolutionary Algorithms (EA) and Swarm Intelligence (SI) algorithms are advantages for solving many real world optimization problems especially when they are with the NP hard characteristics. These algorithms have the anytime behavior by giving a feasible solution at any given time and dealing with multiple solutions at a given time in the predefined solution space. Exploration and exploitation properties embedded in these algorithms allow them to search the reliable solutions in the solution space. This study is consisting of two popular nature inspired algorithms; ant colony systems (ACS) and genetic algorithms (GA). ACS is hybridized with a modified version of GA to find multiple solutions.

We structured the remainder of the paper as follows. The literature related to the multi-model traveling salesmen problem is discussed in Section II. Section III is dedicated to introduce the AGH; the ant genetic hybrid solution we propose and its capabilities. In Section IV, we points out the numerical examples, the information on parameters used in the study and the results obtained. A statistical comparison with existing methods is also included. Finally, we draw conclusions briefly in Section V.

II Solving Discrete Multi-Model Optimization Problems – Summarizing Similar Studies

The research problem discussed in this paper mainly focuses on discrete multi-model optimization, specifically multi-model traveling salesman problem. Multi-model optimization can be defined as an optimization which focuses on finding more than one global and local solutions. Let's say there is a problem to be solved to find minimum/maximum $f(x)$, and the optimum solution is R. Then if there is a possibility of finding several x values which gives you R or there exist only x which is unique but there exist several local optimums (optimal in a certain neighborhood) is then can be stated as a multi-model solution. Therefore you may obtain local or global optimum values by solving a multi-model optimization problem. The problem that we address here belongs to the discrete domain which is the famous traveling salesmen problem having many global optimum routes. The main aim of the hybrid algorithm proposed in this paper is to catch all the global optima in a multi-model traveling salesmen problem.

In practice, there are often problems with multiple solutions. In particular, the traveling salesman problem (TSP) can have several shortest tours from which travelers can choose one, depending on their specific requirements. The travelling salesman problem (TSP) is an NP-hard problem in combinatorial optimization studied in operations research and theoretical computer science. Given a collection of cities and the cost of travel between each pair of them, the traveling salesman problem, is to find the cheapest way of visiting all of the cities and returning to the starting point. In the standard version, the travel costs are symmetric in the sense that traveling from city X to city Y costs just as much as traveling from Y to X. This problem was first formulated as a mathematical problem in 1930 and is one of the most intensively studied problems in optimization. It is used as a benchmark for many optimization methods. There can be many studies related to finding optimal ways to solve single objective and multi-objective TSPs [16]–[20].

Multi-model TSPs is a topic which has been rarely addressed in the world of optimization research. Only handful of research can be found on the specific topic. Here we discuss some findings on multi-model optimization over discrete domains including the MMTSP.

The very first published work on the matter of finding multiple optimum solutions in the discrete domain is dated back to 1995 [21]. The authors addressed the same problem of MMTSP and attempted to solve it using genetic algorithms; what we are adapting in this research as well. As what we concern in the present research, they have also paid attention on finding a way that will get the help of niching strategies at its minimum. Representation method, fitness measures and the evolving mechanisms were presented with a novel idea to solve a MMTSP and a TSP problem was constructed on testing purpose. One disadvantage; although should not mention as such since the time this research has been carried out is more than two decades back, is the prior definition of the number of solutions that expect from the algorithm. Another important fact is that in there, the solutions were obtained in different runs where our approach tries to make it simultaneously which are obviously advantageous. Niching is common in stage when it is about multi-model optimization; no matter the type of the domain. Simply put, niching is a class of methods that try to converge to more than one solution during a single run. Niching is a general class of techniques intended to end up with roughly half the population converging in each minima/maxima. The idea here is that you discourage convergence to a single region of the fitness function by pretending there are limited resources there. The more individuals try to move in, the worse off they all are. The result is that as the GA converges to a single local optimum somewhere, the fitness of that optimum decreases because of the increased competition within the niche. Eventually, another region of the fitness landscape becomes more attractive, and individuals migrate over there. The idea is to reach a steady state - a fixed point in the dynamics - where an appropriate representation of each niche is maintained. This is the most common approach that has been adopted by many of the research conducted on multi-model optimization [5], [15], [22].

In 2006, a research has been carried out to find out how the niching strategies can be adopted to the ant colony optimization algorithms where the initial works of niching strategies can be found with evolutionary algorithms [23]. The TSP known as 'crown' problem was tested with the proposed algorithm. Later in 2018 solving MMTSP problems were addressed with another ant colony system algorithm incorporated with niching methods [24]. One important fact is that in that study, they have designed a test suite for multi-model TSPs which is useful and has been used in our study as well. In a similar study which was carried out at the same time, genetic algorithm has been used incorporating niching methods titled as "neighborhood based genetic algorithm" [25]. The same research group, later in 2020 has published their work of solving multi-model TSPs using a niching memetic algorithm. The test suite with 25 test problems that they have designed in previous study has been used for the testing purposes [26].

The literature points out following important facts on solving a discrete multi-model optimization problem.

- TSP is the most common discrete problem that has been solved as a discrete multi-model optimization problem.
- Very limited approaches can be seen on solving discrete multi-model optimization problems.
- In almost all the studies, a kind of niching strategy has been carried out to find the multiple solutions.

For the convenience, the summary of the findings regarding discrete multi-model optimization are listed in Table 1.

Tab. 1: Summary of the findings regarding discrete multi-model optimization

Year	Paper	Used Algorithm and/ or technique	Niching (Yes/ No)
1995	[21]	Genetic Algorithms + Multiple Solution Technique	Yes
2006	[23]	Ant Colony Optimization algorithm + Fitness Sharing / Simple Crowding	Yes
2018	[25]	Genetic Algorithms + neighborhood-based strategy	Yes
2018	[24]	Ant Colony System + niching strategy and multiple pheromone matrices	Yes
2020	[26]	Memetic Algorithms + niche preservation technique	Yes

However researchers have pointed out some drawbacks of using niching methods on solving multi-model optimization problems such as difficulties in maintaining found solutions, specifying niching parameters, scalability and performance measuring [27]. These findings motivated to carry out the present work to find multiple optimum solutions simultaneously in MMTSPs without using existing niching strategies. We were interested to use latest trends in the field of meta-heuristics; use of hybrid algorithms to solve MMTSPs. We used a crossed version of two famous nature inspired algorithms; ant colony systems algorithm and the genetic algorithm.

III AGH; The ant genetic hybrid solution to solve MMTSPs

Here we discuss the outline of the hybrid algorithm and the full explanation on what we propose for solving multi-model traveling salesman problem. First sections will briefly introduce the two algorithms used; ACS and GA, and the modifications. The latter half presents the detailed description of the new hybrid algorithm that has been proposed with the modifications.

A Ant colony system algorithm (ACS)

Being among the three major extensions of ant colony optimization algorithms, the ant colony system algorithm is one of the most popular and most widely used metaheuristic algorithms in the field of Swarm Intelligence. It is the most recommended and widely used meta-heuristic for the route finding problems [16]. The ACS algorithm is inspired by the optimized routing process from food source to the destination of most ant species, aiming the pheromone communication technique between them. The purpose is to find a good path between the colony and a food source. In the standard ant colony system, this good path refers to the shortest path.

As going along the natural concept, the ACS works as follows. The primary step of the algorithm is to position the artificial ants on random starting points. While they are traveling among the nodes, the route they follow is stored for future usage. For a particular ant, to select the next city from where it is now is decided considering the amount of pheromone deposited in the nodes by the other ants (known as state transition). Each ant, upon visiting a city is responsible for laying some amount of pheromone according to the rule of local pheromone updating. Once all ants complete the tours, the cities followed by the best ant in the population will also benefited with some extra pheromone amount; according to the rule of global pheromone updating. The pseudo code of the algorithm is given in Algorithm 1.

State transition rule is responsible for an ant to find its next visiting city. Assume the ant is in the node r. Its next city s is determined by the equation 1.

$$s = \begin{cases} arg \;\; max_{u \in J_k(r)} \{[\tau(r,u)^{\theta}].[\eta(r,u)]^{\beta}\} & q \leq q_0 \\ S & Otherwise \end{cases} \quad (1)$$

Where, $\tau\,(r,u)$ is the pheromone density of an edge (r,u), $\eta(r,u)$ is [1/distance(r,u)] for TSP. $J_k(r)$ is the set of cities that remain to be visited by ant k positioned on city r. The relative importance of the pheromone trail and the heuristic information are represented by the parameters θ and β $(\theta, \beta \geq 0)$. q is a random number uniformly distributed in [0, 1], q_0 is a parameter$(0 \leq q_0 \leq 1)$, and S is a random variable from the probability distribution given by the equation 2.

$$P_k(r,s) = \begin{cases} \dfrac{[\tau(r,u)]^{\theta}.[\eta(r,u)]^{\beta}}{\sum_{u \in J_k(r)} [\tau(r,u)]^{\theta}.[\eta(r,u)]^{\beta}} & if s \in J_k(r) \\ 0 & Otherwise \end{cases} \quad (2)$$

ACS local and global updating happens according to the equation 3 and equation 4 respectively.

$$\tau(r,s) \leftarrow (1-\rho).\tau(r,s) + \rho.\Delta\tau(r,s) \quad (3)$$

Where 0 < ρ < 1 is a parameter.

$$\tau(r,s) \leftarrow (1-\alpha).\tau(r,s) + \alpha.\Delta\tau(r,s) \quad (4)$$

.

Where

$$\Delta\tau(r,s) = \begin{cases} (L_{gb})^{-1} & if (r,s) \in global \;\; best \;\; tour \\ 0 & Otherwise \end{cases} \quad (5)$$

$0 < \alpha < 1$ is the pheromone decay parameter and L_{gb} is the length of the globally best tour.

Algorithm 1 : Pseudo code of the ACS Algorithm

```
1:  Begin;
2:  Initialize parameter values
3:  Define the objective function
4:  while End condition do
5:      Position ants on starting nodes
6:      for $i = 1 : n$ (for each ant $i$ in the population) do
7:          Build the tour while local pheromone update
            (Equations 1,2,3)
8:      end for
9:      Do global pheromone update to the globally best ant.
        (Equation 4)
10: end while
11: Post process results and visualization;
12: End
```

In the AGH, the ACS works as the initializer by giving a single promising solution to a given MMTSP instance.

B Genetic Algorithms (GA)

GA seems to be the most significant and widely used evolutionary algorithm since its inception. Its flexibility over different domains makes it easy to use and as a result many NP hard problems have their GA version of solutions [28] – [32]. An individual in GA is known as a chromosome which is typically a feasible solution to the problem to be solved. With GA, it can be stated that the binary encoded GA is the most popular version, however many other representations including real value, integer and permutation are also being used. Fitness of a population is calculated and the fitter individuals are then selected probabilistically to be parents to mate and produce offspring. Crossover and mutation are mainly used to generate offspring from the parents providing exploitation and exploration capabilities to the algorithm. Over time, as results of successful research, various crossover and mutation operators for different representations have been presented.

Crossover basically allows new individuals to have blocks of genes from its parents representing the crossover happens in the natural genes. It is mutation that handles the part that cannot be done by this inheritance, allowing the new offspring to be slightly differing from their parents in a probabilistic manner. It further works on the solutions to get out without being trapped in local optimum solutions. The algorithm will be repeated until a predefined termination criterion is satisfied. The general framework of the genetic algorithm is given in Algorithm 2.

Algorithm 2 : Pseudo code of the GA Algorithm

```
1:  Begin;
2:  Initialize algorithm parameters:
3:  Define the objective function
4:  Generate random population of chromosomes
5:  while End Condition do
6:      [Fitness] Evaluate the fitness of each chromosome
7:      while new population is complete do
8:          [Selection] Select parent chromosomes
9:          [Crossover] Perform crossover to form a new off-
            spring
10:         [Mutation] Perform mutation on new offspring
11:         [Accepting] Place new offspring in a new popula-
            tion
12:         [Replace] Use new generated population
13:     end while
14:     Go to step 6
15: end while
16: [Test] Upon completion, return the best solution in cur-
    rent population
17: End;
```

C AGH; The ant, genetic hybrid solution to solve MMTSPs

In order to implement the proposed hybrid, several modifications were implemented on the standard genetic algorithm. The GA with proposed modifications is named as MODGA. The modifications are proposed in order to enhance the exploration capability of the standard genetic algorithm. By adopting the concept of hybridization, we divide the solution into two parts where ACS finds one optimum tour of a TSP and taking that as a threshold, the MODGA finds other
Optimal tours as much as possible. We proposed two modifications to the standard GA, which has been introduced in a previous study for the continuous domain [33]. The concept of archiving and the use of counter variable are included in the modification. The hybrid AGH will work as follows.

- Initially n ants are placed on starting nodes.
- Ants will build tours with local and global pheromone updating rules.
- At the termination, best ant tour with the optimal tour length is selected. (Let K be the optimal tour length given by the ACS).

Using the K (threshold), MODGA algorithm operates as follows.

- Each chromosome (a permutation) in the GA represents a possible solution to the TSP problem.
- Fitness of a chromosome is calculated using the objective function. We used the distance of the route given by the solution as the objective function.

- Then selection, crossover and mutation will perform on the population to generate offspring. We have used 'generational' population model where we generate 'N' offspring, where 'N' is the population size, and the entire population is replaced by the new one at the end of the iteration.

- From the new generation, the better chromosomes were identified (chromosomes whose fitness is ≤ to K), and the archiving concept is used to collect them. The empty positions in the population are then filled with random chromosomes.

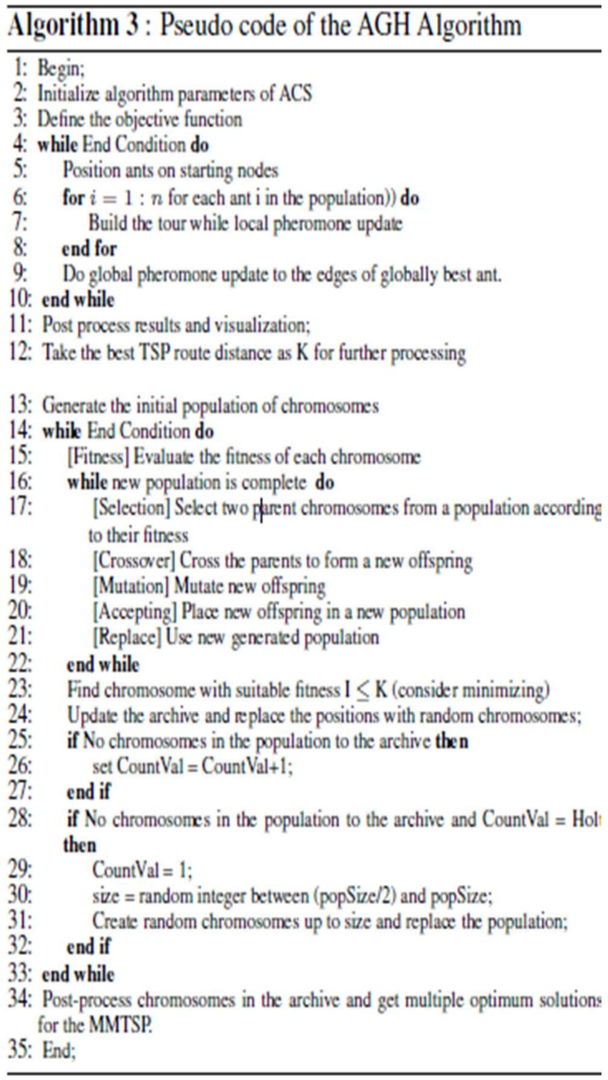
- **A counting variable** is located to identify the poor iterations (iterations which are not contributed to the archive for a specified period (Holt)) and new chromosomes are introduced to the population in a random manner. In the standard GA, crossover and mutation updates a solution using both exploitation and exploration properties. Since our concern lies on enhancing the random replacements when significant performance cannot be seen, this modification further tries to enhance the exploration capability.

- Finally, after a fixed number of iterations, the output of the AGH is the possible optimal routes for the MMTSP which are located in the archive.

For further clarification of the proposed approach, the pseudo-code of the AGH (ACS MODGA Hybrid) to solve MMTSPs is given in Algorithm 3.

It is important to investigate the features of AGH which have contributed to the capability of the algorithm in solving MMTSPs. One of the major decisions that have to be taken is how to identify the global optimum when it is unknown. It is achieved via an approximation done by the ACS algorithm. Once we have an approximation we used it as a threshold to find other possible routes having same, closer or better route distances with the enhanced genetic algorithm. By introducing an archive and the mechanism to identify poor iteration circles, we tried to improve the exploration capability of the algorithm to find multiple solutions as much as possible. Once a solution is found we put it to the archive and replace its position with a random solution in order to change the direction of the search (searching towards the same solution is discouraged). The speed of the algorithm is enhanced with the random replacements introduced for the poor iteration circles (when no improvement can be seen in iterations for a predefined number). These modifications to the standard GA gave acceptable solutions to the MMTSPs.

The significant features of this hybrid can be stated as it uses no niching strategy that has been used in almost all the approaches we have seen to solve multi-model optimization problems. This can be considered as a milestone in the discrete multi-model research where the opportunities are revealed apart the traditional niching methods. The algorithms used in this study are very simple and user friendly and the modifications are simple but precious. The concepts adapted enhance the algorithm by giving more exploration ability and the early identification of poor iterations support to acquire the maximum benefits.


Algorithm 3 : Pseudo code of the AGH Algorithm

```
1:  Begin;
2:  Initialize algorithm parameters of ACS
3:  Define the objective function
4:  while End Condition do
5:      Position ants on starting nodes
6:      for $i = 1 : n$ for each ant i in the population)) do
7:          Build the tour while local pheromone update
8:      end for
9:      Do global pheromone update to the edges of globally best ant.
10: end while
11: Post process results and visualization;
12: Take the best TSP route distance as K for further processing

13: Generate the initial population of chromosomes
14: while End Condition do
15:     [Fitness] Evaluate the fitness of each chromosome
16:     while new population is complete do
17:         [Selection] Select two parent chromosomes from a population according
            to their fitness
18:         [Crossover] Cross the parents to form a new offspring
19:         [Mutation] Mutate new offspring
20:         [Accepting] Place new offspring in a new population
21:         [Replace] Use new generated population
22:     end while
23:     Find chromosome with suitable fitness $I \leq K$ (consider minimizing)
24:     Update the archive and replace the positions with random chromosomes;
25:     if No chromosomes in the population to the archive then
26:         set CountVal = CountVal+1;
27:     end if
28:     if No chromosomes in the population to the archive and CountVal = Hol
        then
29:         CountVal = 1;
30:         size = random integer between (popSize/2) and popSize;
31:         Create random chromosomes up to size and replace the population;
32:     end if
33: end while
34: Post-process chromosomes in the archive and get multiple optimum solution
    for the MMTSP.
35: End;
```

IV Experimentation

AGH can be considered as the very first attempt of using swarm intelligence and evolutionary algorithms to solve a discrete multi-model traveling salesman problem without using a niching method. This section details the test problems and the obtained results of the suggested hybrid.

This research work is carried out on an Intel Core i7 laptop with a RAM of 4GB. Program is developed in MATLAB. Parameter settings of the algorithms are presented in the Table 2.

Tab. 2: Parameter values used in the AGH Algorithm

ACS		MODGA	
Parameter	Value	Parameter	Value
α, ρ (Pheromone decay parameter)	0.1	P_C (Partially Mapped Crossover)	0.85- 0.95
q_0, β	0.9, 2	P_m (Swap mutation)	0.01- 0.25
Population Size	50	PopulationSize	50- 200
Iterations	200	Iterations	500- 2000

A Summary of Benchmark MMTSPs used

Set of MMTSP problems were selected from different studies as well as from the TSPLIB [21], [23]–[26], [34]. Details of the twenty (20) benchmark MMTSPs used are given below.

- **T1**: The TI test problem was devised by selecting towns on the Euclidean plane in such a way that two very different global optima were expected. This MMTSP was originally proposed in [21] and they have found two different global optimum / maximum routes.
- **T2**: The 'crown' problem- The 'crown' problem is a symmetric, 2-Dimensional Euclidean TSP containing 6 cities used in [23]. They were able to locate two different optimum routes.
- The 06 MMSTPs given in Table 3 were taken from [24].

Tab. 3:

Problem	No of cities	Optimal length	# of solutions according to [24]
self8-1	8	187.444	3
self8-2	8	309.436	2
self10-1	10	225.133	2
self10-2	10	236.212	2
test16	16	918.353	9
ulysses16	16	73.9876	10

- The 12 MMSTPs given in Table 4 were taken from [26].

Tab. 4:

Problem	Noof cities	Optimal length	# of solutions according to the standard repository mentioned in[26]
Simple 9	9	680.8311	3
simple2 10	10	1264.4	4
simple3 10	10	832.2031	13
simple4 11	11	803	4
simple5 12	12	754	2
simple6 12	12	845	4
geometry1 10	10	130.821	56
geometry2 12	12	1344	110
geometry3 10	10	72.4033	4
geometry4 10	10	72	4
geometry5 10	10	78.3758	14
geometry6 15	15	130	196

B Results

The Table 5 shows number of multiple optimum routes obtained by the AGH algorithm for the benchmark MMTSP problems. For 5 MMTSP instances, the AGH algorithm has given outstanding number of solutions where other methods have not given. But it should be noted that, for some complicated problems, AGH algorithm has not performed as some of the other methods with niching strategies.

Tab. 5: Number of different optimum solutions obtained

MMTSP Instance	# of different solutions form previous studies and bench- mark details	# of different solutions from AGH algorithm
T1	2	**6**
T2 – CrownProblem	2	**2**
self8-1	3	**5**
self8-2	2	**2**
self10-1	2	**2**
self10-2	2	**4**
test16	9	**10**
ulysses16	10	9
Simple 9	3	**3**
simple2 10	4	**4**
simple3 10	13	**13**
simple4 11	4	**4**
simple5 12	2	**2**
simple6 12	4	**4**
geometry1 10	56	36
geometry2 12	110	3
geometry3 10	4	**6**

geometry4 10	4	3
geometry5 10	14	13
geometry6 15	196	25

PR and **SR** are standard measures proposed by [35] to evaluate the performance of multi-model optimization problems. We have measured the values for the 20 MMTSPs used (see Table 6).

PR is a measurement that calculates the average percentage of the optimums given by an algorithm over all the runs (here we used 50 runs), and **SR** calculates the average percentage of the runs where all global optimum are given by the algorithm. Equation 6 and 7 show the formulas.

$$PR = \frac{\sum_{i=1}^{NR} NPF_i}{NKP * NR} \qquad (6)$$

Where NPF_i denotes the number of global optima found at the end of the i-th run, NKP the number of known global optima, and NR the number of runs.

$$SR = \frac{NSR}{NR} \qquad (7)$$

SR measures the percentage of best runs (a best run is defined as a run where all known global optima tours are found) out of all runs.

Tab.6: PR AND SR VALUES OF ACS-MODGA FORTHE 20 BENCHMARK PROBLEMS

MMTSP Instance	PR Value	SR Value
T1	1	1
T2 – Crown Problem	1	1
self8-1	1	1
self8-2	1	1
self10-1	1	1
self10-2	1	1
test16	0.8888	0.5
ulysses16	0.35	0
Simple 9	1	1
simple2 10	1	1
simple3 10	0.6923	0
simple4 11	1	1
simple5 12	1	1
simple6 12	0.85	0.6
geometry1 10	0.4125	0
geometry2 12	0.0427	0
geometry3 10	1	1
geometry4 10	0.5	0
geometry5 10	0.7857	0
geometry6 15	0.0648	0

70% of better **PR** values and 60% of better **SR** values indicate the success and the capability of the proposed algorithm. It should be admitting that for some complicated problems, zero **SR** indicated that improvements of the **AGH** are essential to find the all possible routes. Improvements to the algorithm and the parameter values is essential and possibilities will give a good impact as AGH is the first algorithm to solve **MMTSPs** without complicated niching strategies.

Graphical representations of some of the solutions obtained can be seen in Figures 1, 2.

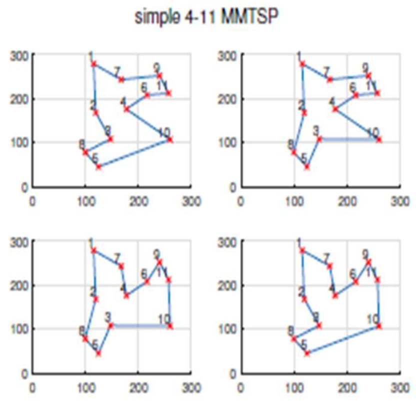


Fig. 1: The four routes obtained by AGH for the ’simple 4-11’ MMTSP instance

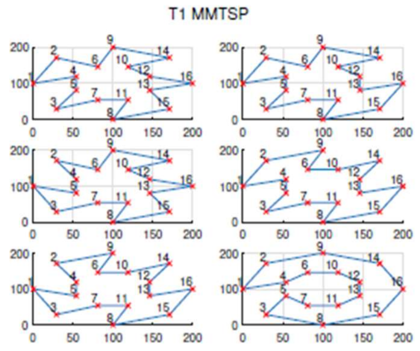


Fig. 2: The six routes obtained by AGH for the ’T1’ MMTSP instance

The graphs on the improvement of the fitness during iterations indicate the viability of the algorithm in optimizing the solutions (see Figure 3).

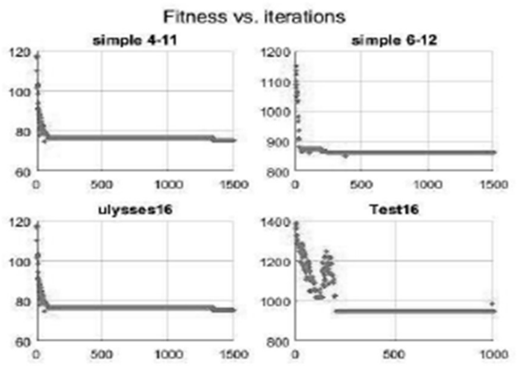


Fig. 3: Fitness changes over iterations of 4 MMTSP instances

For the 12 composite MMTSP problems which includes complicated MMTSPs as well (Table 4), the obtained average distances are as follows (Table 7).

Tab. 7: Average distances obtained by the AGH algorithm

MMTSP Instance	Published Distance	Avg distance from AGH
Simple1 9	680	680.8311
simple2 10	1265	1264.4
simple3 10	832	832.2031
simple4 11	803	803
simple5 12	754	754
simple6 12	845	845
geometry1 10	130	130.821
geometry2 12	1344	1344
geometry3 10	72	72.4033
geometry4 10	72	72
geometry5 10	78	78.3758
geometry6 15	130	130

C Statistical Analysis

A statistical analysis has been done to compare the performance of the **AGH** algorithm with number of solutions given in the standard **MMTSP** repository mentioned in [26]. We used Wilcoxon signed-rank test; a non-parametric test which applies to two related samples [36]. It is an alternative test for the paired Student's t-test or the t-test for dependent samples when the population cannot assume to be normally distributed [37]. It uses the standard normal distributed z value to test of significance. We used results (number of different alternative roots) from **AGH** algorithm with number of solutions given in the standard **MMTSP** repository [26] for the set of 12 **MMTSP**s as two related samples. We conducted the test using **SPSS** statistical software, to test the following hypothesis.

H_0: There is no significant difference between number of solutions given by **AGH** and the standard **MMTSP** repository [26]

H_1: There is a significant difference between number of solutions given by **AGH** and the standard **MMTSP** repository [26]

We received the following output. The SPSS output has given both a W-value and z-value. Since the size of N is low (different values), and particularly it's below 10, we used the W-value to evaluate the hypothesis.

```
Result Details

W-value:  3
Mean Difference:  61
Sum of pos.  ranks:  18
Sum of neg.  ranks:  3

Z-value:  -1.5724
Sample Size  (N):  6
```

The value of W is 3. The critical value for W at N = 6 ($p < 0.05$) is 0. So we do not reject H0; that is we accept that there is no significant difference between the number of solutions given by AGH and the standard MMTSP repository [26].

As a final supposition, we can state that with the conducted statistical analysis, the AGH is performing well for the MMTSPs given in the standard MMTSP repository mentioned in [26]. However, the improvements are significant for most cases. For this reason, we say that the presented AGH is capable in solving MMTSPs, meeting, in this respect, the main objective of this study which is to find a hazel free approach to deal with discrete multi-model optimization problems. It should be noted that AGH is unable to perform well with MMTSPs which have large number of multiple solutions. We suppose that the reason is the less exploration power of MODGA due to the lacking of niching strategies. But with careful modifications, in future we hope to improve AGH; without complicated niching work, to be suitable for such MMTSPs as well.

V Conclusions and further work

Here we presented a hybrid algorithm (AGH) implemented using ACS and GA, to solve multi-model traveling salesmen problem. The proposed algorithm is two phased. ACS finds the initial optimum value for a tour which works as a threshold in the second half. The MODGA which uses an archive and a flag/counter variable runs in the second half with the given threshold to find the multiple optimum solutions/routes for the MMTSP. The algorithm does not relay on niching strategies as all the other algorithms that has been used in the literature to solve MMTSPs. This can be stated as the most significant feature of this algorithm that provides a less complicated approach to solve MMTSPs.

In order to prove the capability of the proposed AGH algorithm, we have compared the performance on more than 20 MMTSPs with the existing methods. A statistical analysis has been carried out on the obtained results using Wilcoxon signed-rank test. Overall, the AGH algorithm performs equally well for many MMTSP instances. However for the instances having large number of different optimum routes, AGH was unable to perform well. We believe that as further enhancements, amplifying the exploration ability of the algorithm will provide better solutions. Here, it is important to highlight that the main goal of this study is not to find an accurate algorithm for MMTSPs, rather to find the capability of implementing a flexible, less complicated algorithm which can handle discrete multi-model optimization problems.

We also expect to consider the parameter tuning of the two algorithms used. As we know, especially in swarm intelligence algorithms, there are many algorithm specific parameters to be fixed by the user. This will directly affect the performance of the algorithm. To popularize the usage of these algorithms, parameter tuning techniques are essential to find the proper parameter values without the involvement of the user. Further, inspired by the results obtained from our experiment; we are planning to expand the study on solving other discrete multi-model optimization problems.

REFERENCES

[1] ‘Multimodal optimization,’ in Introduction to Evolutionary Algorithms. London: Springer London, 2010, pp. 165–191, ISBN: 978-1-84996-129-5. DOI: 10.1007/978- 1- 84996- 129- 5_5. [Online]. Available: https://doi.org/10.1007/ 978-1-84996-129-5_5.

[2] J. Barrera and C. A. C. Coello, ‘A review of particle swarm optimization methods used for multimodal optimization,’ Innovations in swarm intelligence, pp. 9–37, 2009.

[3] B. Boˇskovi´c and J. Brest, ‘Clustering and differential evolution for multimodal optimization,’ in 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2017, pp. 698–705.

[4] K. Deb, ‘Genetic algorithms in multimodal function optimization,’ Ph.D. dissertation, Clearinghouse for Genetic Algorithms, Department of Engineering Mechanics, 1989.

[5] N. Glibovets and N. Gulayeva, ‘A review of niching genetic algorithms for multimodal function optimization,’ Cybernetics and Systems Analysis, vol. 49, no. 6, pp. 815–820, 2013.

[6] J. K. Lenstra and A. R. Kan, ‘Some simple applications of the travelling salesman problem,’ Journal of the Operational Research Society, vol. 26, no. 4, pp. 717–733, 1975.

[7] B. Gavish and S. C. Graves, ‘The travelling salesman problem and related problems,’ 1978.

[8] G. Laporte, A. Asef-Vaziri and C. Sriskandarajah, ‘Some applications of the generalized travelling salesman problem,’ Journal of the Operational Research Society, vol. 47, no. 12, pp. 1461–1467, 1996.

[9] P. Siarry, A. P´etrowski and M. Bessaou, ‘Amultipopulation genetic algorithm aimed at multimodal optimization,’ Advances in Engineering Software, vol. 33, no. 4, pp. 207–213, 2002.

[10] B.-Y. Qu, P. N. Suganthan and J.-J. Liang, ‘Differential evolution with neighborhood mutation for multimodal optimization,’ IEEE transactions on evolutionary computation, vol. 16, no. 5, pp. 601–614, 2012.

[11] R. Thomsen, ‘Multimodal optimization using crowding based differential evolution,’ in Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753), IEEE, vol. 2, 2004, pp. 1382–1389.

[12] M. Preuss, Multimodal optimization by means of evolutionary algorithms. Springer, 2015.

[13] P. D. Justesen, ‘Multi-objective optimization using evolutionary algorithms,’ University of Aarhus, Department of Computer Science, Denmark, vol. 33, 2009.

[14] J. R¨onkk¨onen et al., ContinuousMultimodal Global Optimization with Differential Evolution-Based Methods. Lappeenranta University of Technology, 2009.

[15] B. L. Miller and M. J. Shaw, ‘Genetic algorithms with dynamic niche sharing for multimodal function optimization,’ in Proceedings of IEEE international conference on evolutionary computation, IEEE, 1996, pp. 786–791.

[16] M. Dorigo and L. M. Gambardella, ‘Ant colony system: A cooperative learning approach to the traveling salesman problem,’ IEEE Transactions on evolutionary computation, vol. 1, no. 1, pp. 53–66, 1997.

[17] P. Larranaga, C. M. H. Kuijpers, R. H. Murga, I. Inza and S. Dizdarevic, ‘Genetic algorithms for the travelling salesman problem: A review of representations and operators,’ Artificial Intelligence Review, vol. 13, no. 2, pp. 129–170, 1999.

[18] R. L. Karg and G. L. Thompson, ‘A heuristic approach to solving travelling salesman problems,’ Management science, vol. 10, no. 2, pp. 225–248, 1964.

[19] D. Angus, ‘Crowding population-based ant colony optimization for the multi-objective travelling salesman problem,’ in 2007 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision- Making, IEEE, 2007, pp. 333–340.

[20] C. Changdar, G. Mahapatra and R. K. Pal, ‘An efficient genetic algorithm for multi-objective solid travelling salesman problem under fuzziness,’ Swarm and Evolutionary Computation, vol. 15, pp. 27–37, 2014.

[21] S. Ronald, ‘Finding multiple solutions with an evolutionary algorithm,’ in Proceedings of 1995 IEEE International Conference on Evolutionary Computation,

IEEE, vol. 2, 1995, pp. 641–646.

[22] X. Lin, W. Luo and P. Xu, ‘Differential evolution for multimodal optimization with species by nearest better clustering,’ IEEE transactions on cybernetics, 2019.

[23] D. Angus, ‘Niching for population-based ant colony optimization,’ in 2006 Second IEEE International Conference on e-Science and Grid Computing (e-Science’06), IEEE, 2006, pp. 115–115.

[24] X.-C. Han, H.-W. Ke, Y.-J. Gong, Y. Lin, W.-L. Liu and J. Zhang, ‘Multimodal optimization of traveling salesman problem: A niching ant colony system,’ in Proceedings of the Genetic and Evolutionary Computation Conference Companion, 2018, pp. 87–88.

[25] T. Huang, Y.-J. Gong and J. Zhang, ‘Seeking multiple solutions of combinatorial optimization problems: A proof of principle study,’ in 2018 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2018, pp. 1212–1218.

[26] T. Huang, Y.-J. Gong, S. Kwong, H. Wang and J. Zhang, ‘A niching memetic algorithm for multisolution traveling salesman problem,’ IEEE Transactions on Evolutionary Computation, vol. 24, no. 3, pp. 508–522, 2019.

[27] X. Li, M. G. Epitropakis, K. Deb and A. Engelbrecht, ‘Seeking multiple solutions: An updated survey on niching methods and their applications,’ IEEE Transactions on Evolutionary Computation, vol. 21, no. 4, pp. 518–538, 2016.

[28] C. R. Houck, J. Joines and M. G. Kay, ‘A genetic algorithm for function optimization: A matlab implementation,’ Ncsu-ie tr, vol. 95, no. 09, pp. 1–10, 1995.

[29] G. C. Dandy, A. R. Simpson and L. J. Murphy, ‘An improved genetic algorithm for pipe network optimization,’ Water resources research, vol. 32, no. 2, pp. 449–458, 1996.

[30] J. Horn, N. Nafpliotis and D. E. Goldberg, ‘A niched pareto genetic algorithm for multi objective optimization, ‘in Proceedings of the first IEEE conference on evolutionary computation. IEEE world congress on computational intelligence, Ieee, 1994, pp. 82–87.

[31] P. C. Chu and J. E. Beasley, ‘A genetic algorithm for the multidimensional knapsack problem,’ Journal of heuristics, vol. 4, no. 1, pp. 63–86, 1998.

[32] F. H. Khan, N. Khan, S. Inayatullah and S. T. Nizami, ‘Solving tsp problem by using genetic algorithm,’International Journal of Basic & Applied Sciences, vol. 9, no. 10, pp. 79–88, 2009.

[33] M. Ariyaratne, T. Fernando and S Weerakoon, ‘A hybrid algorithm to solve multi-model optimization problems based on the particle swarm optimization with a modified firefly algorithm,’ in Proceedings of the Future Technologies Conference, Springer, 2020, pp. 308–325.

[34] G. Reinelt, ‘TSPLIB–a traveling salesman problem library,’ ORSA Journal on Computing, vol. 3, no. 4, pp. 376 384, 1991.

[35] X. Li, A. Engelbrecht and M. G. Epitropakis, ‘Benchmark functions for cec’2013 special session and competition on niching methods for multimodal function optimization,’ RMIT University, Evolutionary Computation and Machine Learning Group, Australia, Tech. Rep, 2013.

[36] F. Wilcoxon, ‘Individual comparisons by ranking methods,’ in Breakthroughs in statistics, Springer, 1992, pp. 196–202.

[37] M. Hollander, D. A. Wolfe and E. Chicken, Nonparametric statistical methods. John Wiley & Sons, 2013, vol. 751.