Int. J. Nonlinear Anal. Appl. 14 (2023) 1, 275–285
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2022.26610.3364

Solving NP hard problems using a new genetic algorithm

Mohammad Ali Ebrahimi[a], Hassan Dehghan Dehnavi[a,*], Mohammad Mirabi[b], Mohammad Taghi Honari[a], Abolfazl Sadeghian[a]

[a] *Department of Industrial Management, Yazd Branch, Islamic Azad University, Yazd, Iran*

[b] *Department of Industrial Engineering, Meybod University, Meybod, Iran*

(Communicated by Mohammad Bagher Ghaemi)

Abstract

Over the past few decades, a lot of meta-heuristics have been developed to solve N-P hard problems. Genetic algorithm, ant colony optimization, simulated annealing, electromagnetism algorithm and tabu search are some examples of meta-heuristics algorithms. These kinds of algorithms have two main classes: population-based and Trajectory. Many of these algorithms are inspired by various phenomena of nature. In this research, the author introduces a new population-based method inspired by the lifestyle of lions and the genetic algorithm's structure called the new genetic algorithm (NGA). The social behaviour of lions and genetic operators like mutation and cross-over is the main structure of NGA. Finally, the NGA is compared with the hybrid genetic and hybrid ant colony optimization as the best existing algorithms in the literature. The experimental results have revealed that the NGA is competitive in terms of solution quality to solve the vehicle routing and scheduling problems as two main categories of N-P hard problems.

Keywords: New genetic algorithm, N-P Hard problem, Scheduling, Vehicle routing problem, Genetic operator, Ant colony optimization.
2010 MSC: 68T20, 68W50

1 Introduction

Problems can be categorized into two main classes P and NP based on the computational theory. The problem of the complexity class of P can be solved by a deterministic algorithm in polynomial time. Then it is relatively easy to solve. Minimum spanning tree, shortest path problems, maximum flow network, maximum bipartite matching, and linear programming continuous models belong to the P class. The problem of the complexity class of NP can be solved by a nondeterministic algorithm in polynomial time.

A decision problem belonging to NP is NP-complete if all other problems of class NP are reduced in polynomial time. NP-hard problems are optimization problems whose associated decision problems are NP-complete. Most of the real-world optimization problems and many academic popular problems are NP-hard. Routing and covering problems such as vehicle routing problems (VRP), Sequencing and scheduling problems such as flow-shop scheduling, job-shop scheduling, Knapsack and packing/cutting problems and Assignment and location problems such as quadratic assignment problem (QAP) are some cases of NP-hard problems [13].

*Corresponding author
Email addresses: mama.ebrahimi200@gmail.com (Mohammad Ali Ebrahimi), denavi2000@yahoo.com (Hassan Dehghan Dehnavi), mirabi@meybod.ac.ir (Mohammad Mirabi), mthonari@iauyazd.ac.ir (Mohammad Taghi Honari), sadeghian@iauyazd.ac.ir (Abolfazl Sadeghian)

Received: January 2022 *Accepted:* March 2022

For NP-hard problems, provably efficient algorithms do not exist and therefore meta-heuristics in pure and hybrid structures have wide applications to solve this kind of problem.

A meta-heuristic is described as an iterative generation process that guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information to find efficiently near-optimal solutions [24].

In this paper, we introduce a population-based method called a new genetic algorithm (NGA) to solve NP-hard problems. The rest of the paper is organized as follows: Section 2 introduces some efficient meta-heuristics developed to solve NP-hard problems basically in VRP and scheduling area. The NGA is defined in section 3. Section 4 devotes to verification and comparison study and finally section 5 concludes the research.

2 Meta-heuristics

Based on Blum and Andrea [3], meta-heuristics have some basic properties as follows:

- Meta-heuristics are strategies that ”guide” the search process.
- The goal is to efficiently explore the search space to find near-optimal solutions.
- Techniques which constitute meta-heuristic algorithms range from simple local search rules to complex learning processes.
- Meta-heuristic algorithms are approximate and usually non-deterministic.
- They may incorporate mechanisms to avoid getting trapped in confined areas of the search space.
- The basic concepts of meta-heuristics let an abstract level description.
- Meta-heuristics are not problem-specific.
- Meta-heuristics may make use of domain-specific knowledge in the form of heuristics that are controlled by the upper-level strategy.
- Today’s more advanced meta-heuristics use search experience (embodied in some form of memory) to guide the search.

Two main categories of meta-heuristics are:

1. Trajectory methods like basic local search, simulated annealing, tabu search and variable neighbourhood search.
2. Population-based methods like genetic algorithm, ant colony optimization and particle swarm optimization.

Meta-heuristics in both trajectory and population categories are a good choice to solve the NP-hard problems like scheduling and vehicle routing problems (VRP).

2.1 Scheduling

Scheduling is an optimization problem in computer science and operations research in which ideal jobs are assigned to resources at particular times. Scheduling in many environments (flow-shop, job-shop, open-shop and etc.) is known as an NP-hard problem and many researchers have attacked this area by meta-heuristics. In this research, we focus on flow-shop.

One of the first pieces of research on the permutation flow-shop problem (PFSP) is due by Johnson [19] that proposed a simple rule to obtain optimal sequences with two machines. PFSP is NP-complete [14], therefore researchers have mainly focused on effective heuristics and meta-heuristics to solve it. Salient heuristics due to Campbell et al. [4] (also called the CDS method) and the well-known NEH heuristic by Nawaz et al. [23] are some of the first heuristics in this area. Meta-heuristics have also been attended like the simulated annealing by Osman and Potts [25], the tabu search of Widmer and Hertz [36], and the genetic algorithm of Reeves [29].

Ruiz et al. [31] developed two heuristics for the PFSP that outperform some previous research. Ruiz and Stutzle [32] introduced two simple local searches based on the iterated greedy algorithm that performed better than those of Ruiz et al [31]. Benkalai et al. [2] addressed the problem of scheduling a set of independent jobs with set-up times on a set of machines in a permutation flow shop environment. A meta-heuristic known as the Migrating Birds Optimization (MBO) was adapted for the minimization of the make-span in PFSP. Two versions of the algorithm are presented. Jin and Price [18] proposed a new meta-heuristic algorithm that is based on a new enhanced destruction and construction method and a novel repair method while adopting the architecture of the iterated greedy algorithm for the PFSP. Pan et al. [26] addressed the DPFSP with the total flow-time criterion. To suit the needs of different CPU time demands and solution quality, they presented three constructive heuristics and four meta-heuristics. Among all methods, one of the most successful meta-heuristics to solve PFSP is a genetic algorithm like the old researches worked by Reeves [29] and Sun and Hwang [34] and new ones worked by Mirabi [21], Zhang et al. [37] and Cui et al. [9].

Mirabi [21] developed one novel hybrid genetic algorithm (HGA) to solve PFSP. Proposed HGA used a modified method to generate the initial solutions and applied an improved heuristic for improvement. Also, the author used three genetic operators to make good new offspring. He compared HGA to some other heuristics and showed the competitiveness of the proposed HGA. Zhang et al. [37] proposed a hybrid ant colony optimization algorithm to solve the PFSP. The hybridization of ant colony optimization (ACO) with path relinking (PR), which combines the advantages of two individual algorithms, is the key creative aspect of their research.

2.2 Vehicle routing problem

The vehicle routing problem (VRP) is a combinatorial optimization and integer programming problem which search for the optimal set of routes for a fleet of vehicles that traverse to deliver a given set of customers. The vehicles leave the depot, serve customers in the network, and on completion of their routes return to the depot.

In the recent fifty years since Dantzig and Ramser's [11] research on this area has exploded dramatically. The VRP has extensive variants, including green VRP, periodic VRP (PVRP, where the customers are served periodically), VRP with pickup and delivery (VRPPD, where the customers may both receive and send products), VRP with time windows (VRPTW, where the vehicles must arrive at the customers before the latest arrival time while arriving before the earliest arrival time results in waiting), multi-depot VRP (MDVRP, where more than one depot is considered), multi-depot PVRP (MDPVRP), and so on [11].

Da Costa et al. [10] proposed a Genetic Algorithm (GA) to address a Green Vehicle Routing Problem. Drummond et al. [12] developed an island-based parallel evolutionary method for the PVRP, which evolves individuals representing schedules (patterns), the fitness being obtained by constructing routes for each period with a saving heuristic. In 2007, Alegre et al. [1] presented a scatter search procedure just for solving PVRP with many periods.

One challenging point in PVRPs is local optimum. Chao et al. [5] used a local search approach to overcome this situation and showed it works efficiently when there is no vehicle capacity constraint. Another heuristic that was used by Cordeau et al. [7] was Tabu search in a basic and unified format. Hemmelmayr et al. [16] applied the variable neighbourhood search and outperformed previous approaches.

For the MDVRP, geometric aspects have been attended by a few researchers. Mutation operator introduced by Thangiah and Salhi [35]. They target the depot assignment to "borderline" customers, which are near several depots. In this regard, some approaches are more efficient, such as methods based on neighbourhood search like the Tabu search algorithms [6, 30] and simulated annealing [20]. Pisinger and Ropke [27] have attended the MDVRP by an adaptive large neighbourhood search method, which implements the ruin and recreate paradigm with adaptive selection operators. Ho et al. [17] presented two hybrid genetic algorithms (HGAs) for MDVRP. The major difference between the HGAs was the initial solutions. Giosa et al. [15] developed new assignment algorithms for the MDVRP, and Crevier et al. [8] addressed an extension of the MDVRP in which vehicles may be replenished at intermediate depots along their routes. Stodola [33] deals with the modified MDVRP. The modification consisted of altering the optimization criterion. The optimization criterion of the standard MDVRP is to minimize the total sum of routes of all vehicles, whereas the criterion of modified MDVRP (M-MDVRP) is to minimize the longest route of all vehicles.

Mirabi [22] provided a new definition of periodic vehicle routing problems with single and multi-depots and also developed a novel hybrid genetic algorithm to solve it. The proposed hybrid genetic algorithm applies a modified approach to generate a population of initial chromosomes and uses an improved heuristic called the iterated swap procedure to improve the initial solutions.

Reed et al. [28] demonstrated the use of the Ant Colony System (ACS) to solve the capacitated vehicle routing problem associated with the collection of recycling waste from households, treated as nodes in a spatial network.

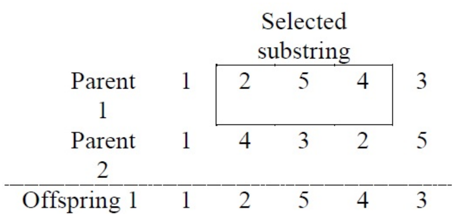
3 The New Genetic Algorithm

Unlike other cats, lions live in groups, called pride. One pride consists of one male as a commander, up to three males, related females, and their cubs. The size of pride depends on available food and water. Fewer resources result in smaller pride. Commander is the strongest lion within the pride and is responsible for guarding their territory and their cubs. He plays this role till another strong male defeats him and gains his pride. Female lions are the primary hunters of the group.

They give birth to 2-3 cubs at a time. Male cubs must leave the pride at around two years old. They form small groups until they are strong enough to challenge male lions of other pride.

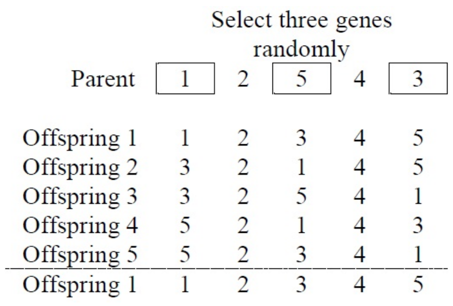
This kind of treatment stimulates us to develop one approach called a new genetic algorithm (NGA) to solve a wide range of problems. First of all, NGA needs an initial population of solutions therefore it is a population-based one. After that, all solutions are divided into some groups (pride). Solutions for each group must be more than 2. The best solution for each group (based on the objective function), is called commander. In the next steps, new solutions (offspring) are generated by using a heuristic mutation operation or order crossover operation in each group.

We use heuristic mutation and order crossover operations like the ones done by Mirabi [21]. Figure 1 and 2 describe these operations.



		Selected substring			
Parent 1	1	2	5	4	3
Parent 2	1	4	3	2	5
Offspring 1	1	2	5	4	3

Figure 1: The order crossover operator



		Select three genes randomly			
Parent	1	2	5	4	3
Offspring 1	1	2	3	4	5
Offspring 2	3	2	1	4	5
Offspring 3	3	2	5	4	1
Offspring 4	5	2	1	4	3
Offspring 5	5	2	3	4	1
Offspring 1	1	2	3	4	5

Figure 2: The heuristic mutation operator

Each new solution in each group challenges all commanders and if one defeats one of them (is better than it), becomes the new commander of the related group and the previous commander and some worst solutions in the group are eliminated.

Let us use the following notations for our algorithm:

m : number of all initial solutions (all populations of lions).

m must be more than 4 because we need at least two groups with a size of at least 2.

n : number of groups (pride). Each group has at least two solutions and therefore $2 \leq n \leq [m/2]$.

k : number of iterations that we need to generate new solutions in each group that challenge other commanders to replacement.

The pseudo-code of NGA is as follows:

```
Input $m$, $n$ and $k$
Describe fitness function (objective function)
Generated $m$ initial solutions by random
Generate $n$ pride numbered 1 to $n$
```

```
For $j = 1 : [m/n] + 1$
  For $i = 1 : n$
    If there are unallocated solution,
      Select one solution by random and allocate it to pride numbered $i$
    End if
  End for
End for
For $i = 1 : n$
    Select the best solution in pride $i$ based on the fitness function and called it as commander (commander=arg max (fitness))
End for
For $i = 1 : k$
  For $j = 1 : n$
    Generate a new solutions in pride $j$ randomly by heuristic mutation of commander or order crossover between commander and one member of pride.
    Compare the new solution with all commanders, if new solution is better than one of them (based on fitness function) replace it with the related commander in the pride and drop previous commander and $[m/3n]$ solutions with the worst fitness function value.
  End for
End for
Compare all commanders and select the best for the final solution.
```

4 Comparison Study

In this section, a computational study is carried out to compare the NGA with the best-developed heuristics in both flow-shop scheduling and the VRP area. For flow-shop, two heuristics as follows were selected:

Hybrid genetic algorithm (HGA) developed by Mirabi [21]

Hybrid ant colony (HACO) developed by Zhang et al. [37]

and for VRP, the following methods were selected for comparison:

Hybrid genetic algorithm (HGA) developed by Mirabi [22]

Ant colony optimization (ACO) developed by Reed et al. [28]

For flow-shop scheduling, we classify the problem based on the number of jobs (N) and the number of machine (M). We consider 28 classes of problems. In each class, 10 instances are randomly generated. Processing and set up time are $U(1, 99)$ and $U(1, 9)$ respectively. Make-span is considered a fitness function and for the evaluation study, we use the PM index defined as follows:

$$PM = \frac{Heu_{sol} - Best_{sol}}{Best_{sol}}, \tag{4.1}$$

where the make-span obtained by a given algorithm is Heu_{sol} and $Best_{sol}$ is the make-span of the best solution obtained by all algorithms. The programs are coded in MATLAB. The standard approach in the experimental comparison of evolutionary algorithms is to repeat several runs on the same problem because of the stochastic nature of the algorithms. For equal conditions between three methods, all algorithms are run 10 independent times with a stopping criterion based on a finite number of iterations.

In Table 1 Min, Max and the average PM of each method are shown. Also, the average time to solve 10 instances is given for each method. The columns labelled "Min" show, in subscript, the number of instances for which the algorithm solution was equal to the corresponding $Best_{sol}$. As shown in Table 1, New Genetic Algorithm (NGA) outperforms others based on PM value and HGA is a bit faster than NGA.

Table 1: PM values for comparison studies between all algorithms (times are in second)

Class of problem	N	M	NGA				HGA				HACO			
			Min PM	Average		Max PM	Min PM	Average		Max PM	Min PM	Average		Max PM
				PM	Time			PM	Time			PM	Time	
1	10	5	0_5	0.06	1.05	0.13	0_4	0.00	1.83	0.01	0_2	0.21	1.96	0.19
2	10	10	0_5	0.05	1.21	0.10	0_3	0.08	1.84	0.14	0_2	0.18	2.77	0.16
3	10	15	0_5	0.05	2.39	0.11	0_5	0.01	2.78	0.01	0.01	0.04	5.53	0.21
4	10	20	0_6	0.03	5.11	0.09	0_3	0.03	4.01	0.05	0_2	0.09	8.28	0.08
5	20	5	0_7	0.00	1.55	0.00	0_2	0.01	2.15	0.02	0_1	0.00	2.44	0.00
6	20	10	0_9	0.01	1.76	0.10	0_3	0.09	2.77	0.17	0_2	0.11	4.79	0.10
7	20	15	0_9	0.02	3.86	0.16	0_4	0.03	2.94	0.06	0.00	0.01	9.56	0.07
8	20	20	0_6	0.02	6.87	0.05	0_5	0.07	5.05	0.17	0_1	0.30	13.31	0.24
9	30	5	0_6	0.05	1.77	0.13	0_3	0.09	2.76	0.13	0_1	0.20	3.43	0.16
10	30	10	0_9	0.02	3.22	0.14	0_3	0.02	3.59	0.03	0_1	0.23	6.85	0.19
11	30	15	0_9	0.00	4.48	0.01	0_2	0.02	4.34	0.02	0.01	0.02	9.25	0.03
12	30	20	0_8	0.00	7.00	0.01	0_5	0.04	6.10	0.10	0.01	0.04	13.53	0.21
13	40	5	0_5	0.04	2.20	0.09	0_4	0.06	3.24	0.11	0_2	0.23	3.40	0.21
14	40	10	0_6	0.07	2.86	0.18	0_4	0.10	4.29	0.13	0.01	0.03	6.62	0.17
15	40	15	0_6	0.00	4.37	0.00	0_4	0.10	5.61	0.18	0.00	0.03	8.11	0.21
16	40	20	0_7	0.03	9.41	0.08	0_3	0.01	7.83	0.01	0.01	0.04	15.62	0.26
17	50	5	0_7	0.03	2.92	0.10	0_3	0.11	3.87	0.17	0_2	0.04	3.40	0.03
18	50	10	0_7	0.03	5.47	0.09	0_4	0.01	5.11	0.02	0.00	0.03	6.63	0.17
19	50	15	0_6	0.02	8.94	0.05	0_4	0.02	7.04	0.04	0.00	0.03	10.84	0.26
20	50	20	0_5	0.08	11.15	0.15	0_5	0.00	10.89	0.00	0.00	0.03	14.86	0.20
21	100	5	0_8	0.03	4.50	0.17	0_5	0.03	6.14	0.08	0_1	0.19	5.96	0.15
22	100	10	0_5	0.05	9.64	0.10	0_5	0.03	13.22	0.08	0.00	0.02	19.22	0.21
23	100	15	0_7	0.04	15.38	0.13	0_3	0.07	18.59	0.13	0.01	0.04	22.94	0.17
24	100	20	0_6	0.01	28.27	0.01	0_3	0.01	27.49	0.01	0_1	0.11	32.28	0.09
25	200	5	0_9	0.01	11.22	0.06	0_4	0.04	14.55	0.08	0.00	0.02	13.66	0.03
26	200	10	0_6	0.03	32.54	0.07	0_4	0.06	26.79	0.13	0.01	0.04	27.74	0.26
27	200	15	0_6	0.00	37.36	0.00	0_5	0.05	32.15	0.15	0.01	0.04	38.28	0.27
28	200	20	0_9	0.01	75.78	0.07	0_2	0.10	73.32	0.14	0_2	0.15	97.60	0.14

At this stage, the ANOVA test can be applied to show whether the results gained by all methods are statistically similar or not. The ANOVA procedure tests these hypotheses:

$H0: \ \mu 1 = \mu 2 = \mu 3 = \mu 4$, all results are the same

$H1$: two or more results are different from the others

With the $\alpha = 0.05$ significance level, computations are shown in Table 2.

Table 2: ANOVA test for all methods

	Some of Square(SS)	Degree of freedom(df)	Mean Square(MS)	VR	F
Between groups (or "Factor")	69828179.5	3	23276059.8	10.5	2.7
Within groups (or "Error")	239874388.7	108	2221059.2		
Total	309702568.2	111			

In Table 2 $VR = 10.5 > F = 2.7$, therefore the results in Table 1 are not the same and differences are statistically significant. Also, Table 1 demonstrates that NGA and HGA are more competitors. For a detailed comparison between NGA and HGA, we should check whether the differences between the solutions of the two algorithms are statistically significant or not. For this, the hypothesis that the population corresponding to the differences has a mean (μ) zero can be tested; specifically, test the (null) hypothesis $\mu = 0$ against the alternative $\mu > 0$. This test is performed like what was done by Mirabi [21] between the two best methods based on Table 1 (NGA and HGA). It is assumed that the differences between solutions (makespan) are a Normal variable, and choose the significance level $\alpha = 0.05$. If the hypothesis is true, the random variable $T = (\bar{X}_1 - \bar{X}_2)/\sqrt{(S_1^2/n_1) + (S_2^2/n_2)}$ has a t distribution with:

$v = (S_1^2/n_1 + (S_2^2/n_2)^2 / \left(\frac{(S_1^2/n_1)^2}{n_1-1} + \frac{(S_2^2/n_2)^2}{n_2-1}\right)$ degrees of freedom. The critical value of c is obtained from the relation Prob $(T > c) = \alpha = 0.05$. Table 3 shown this study. For more explanation, consider the first row of Table 2, corresponds to the sample size= $n_1 = n_2 = 10$, $\mu_0 = 0$, sample mean for NGA and HGA are $\bar{X}_1 = 759.38$ and $\bar{X}_2 = 766.46$ respectively. Sample standard deviation for HGA and H3 are $S_1 = 2.88$ and $S_2 = 2.55$ respectively. Since $t = 1.73 < T = 5.81$, we conclude that the difference is statistically significant.

Table 3: Detail comparison between NGA and HGA

Class of problem	N	M	Ave. MS or ($\bar{X}$)		Ave. SD or (S)		T	v	t	sig.
			NGA	HGA	NGA	HGA				
1	10	5	723.30	695.90	2.69	2.72	-22.66	18	1.73	Yes
2	10	10	1172.94	1134.85	2.14	1.90	-42.05	18	1.73	Yes
3	10	15	1154.80	1198.24	2.20	2.05	45.63	18	1.73	Yes
4	10	20	1478.15	1465.95	4.71	3.81	-6.37	17	1.74	Yes
5	20	5	1227.32	1288.74	3.49	3.39	39.90	18	1.73	Yes
6	20	10	1783.45	1715.23	4.32	4.44	-34.81	18	1.73	Yes
7	20	15	1763.38	1849.41	2.01	2.78	79.30	16	1.75	Yes
8	20	20	2059.69	2199.18	3.80	4.45	75.40	18	1.73	Yes
9	30	5	1744.00	1848.11	4.33	3.69	57.92	18	1.73	Yes
10	30	10	2202.29	2114.29	4.04	4.16	-48.02	18	1.73	Yes
11	30	15	2546.58	2639.54	4.04	4.15	50.72	18	1.73	Yes
12	30	20	1992.28	2181.83	5.54	5.68	75.55	18	1.73	Yes
13	40	5	1922.68	1964.97	5.44	4.85	18.35	18	1.73	Yes
14	40	10	2268.54	2391.76	3.11	3.80	79.37	17	1.74	Yes
15	40	15	2467.59	2533.96	2.94	2.67	52.85	18	1.73	Yes
16	40	20	2770.88	2880.01	5.86	6.11	40.76	18	1.73	Yes
17	50	5	2484.28	2561.98	3.58	5.02	39.86	16	1.75	Yes
18	50	10	2461.20	2652.10	6.73	7.21	61.18	18	1.73	Yes
19	50	15	3011.93	2839.50	7.60	8.31	-48.44	18	1.73	Yes
20	50	20	3455.95	3619.79	7.45	7.42	49.29	18	1.73	Yes
21	100	5	5136.03	4926.06	6.77	6.53	-70.55	18	1.73	Yes
22	100	10	5236.74	5402.69	6.93	7.65	50.83	18	1.73	Yes
23	100	15	5697.40	5890.75	9.59	12.26	39.29	17	1.74	Yes
24	100	20	5728.00	6011.89	10.12	11.04	59.93	18	1.73	Yes
25	200	5	9171.59	9497.92	9.84	10.36	72.19	18	1.73	Yes
26	200	10	10205.33	10317.50	20.76	24.22	11.12	18	1.73	Yes
27	200	15	9195.35	9488.75	15.10	14.62	44.15	18	1.73	Yes
28	200	20	9285.53	10419.70	17.20	19.23	139.01	18	1.73	No
29	400	5	20046.24	21173.59	21.42	21.16	118.38	18	1.73	Yes
30	400	10	19248.25	21429.31	27.42	24.13	188.83	18	1.73	Yes
31	400	15	26718.48	26183.87	31.37	30.01	-38.94	18	1.73	Yes
32	400	20	24331.50	25861.36	21.10	26.47	142.91	17	1.74	Yes
33	400	50	30101.09	31934.58	25.86	28.53	150.59	18	1.73	Yes

Ave:Average, MS:Makespan, SD:Standard deviation, Sig:Significant

Each class contains 10 independent instances

Table 3 demonstrated that NGA outperformed HGA in 76% of all classes and all of the differences are statistically significant. Also, HGA outperformed NGA in 24% of all classes and in all cases, differences are statistically significant.

To do a deep comparison between NGA and HGA Tukey honestly significance difference test can be used. It is a strong statistical tool to check significance by computing the confidence interval similarly to the confidence interval for the difference of two means, but using the q distribution which avoids the problem of inflating α:

$$\bar{x}_i - \bar{x}_j \pm q(\alpha, r, df_w)\sqrt{\frac{MS_w}{2} \times \left(\frac{1}{n_i} + \frac{1}{n_j}\right)}$$

Table 4 summarized the outputs of this test.

Table 4: Tukey test results for NGA and HGA

	$\bar{x}_{HGA} - \bar{x}_{LA}$	**Critical q** $q(\alpha, r, df_w)$	95% Conf Interval for $\mu_{HGA} - \mu_{LA}$		**Significant at 0.05?**
			Min	Max	
$HGA - LA$	120.83	3.90	-89.06	89.11	Yes

Table 4 demonstrated that NGA completely outperforms HGA.

For VRP we considered two major classes single depot (SD) and two depots (multi depot or MD). Service time and independent work time for each customer are assumed to be random integers from uniform distributions [10, 15] and [30, 60] respectively. Two problem characteristics are the number of vehicles and the number of customers. Each class of problem has 1, 3 or 5 vehicles and 20, 50 or 100 customers.

For each problem instance, 10 runs are done. As an example of the notation, consider the MD520 set of problems: it consists of 10 runs of problem instances with two depots, 5 vehicles and 20 customers. All algorithms are stopped at the same CPU time. The value of this CPU time for each class in second is calculated as follows:

$$\text{(number of customer)} * \text{(number of vehicle)}/10$$

Table 5 shows comparison results.

Table 5: PM values for computational test of all methods

Instances	**NGA**			**IACO**			**HGA**		
	PM value			PM value			PM value		
	Min	Ave.	Max	Min	Ave.	Max	Min	Ave.	Max
SD120	0_6	0.02	0.06	0_4	0.06	0.12	0.03	0.06	0.10
SD150	0_5	0.03	0.07	0_4	0.08	0.15	0.08	0.16	0.21
SD1100	0_8	0.01	0.05	0_3	0.10	0.16	0_1	0.05	0.26
SD320	0_8	0.01	0.05	0_2	0.10	0.15	0.04	0.10	0.20
SD350	0_6	0.04	0.10	0_4	0.07	0.15	0_2	0.06	0.10
SD3100	0_5	0.04	0.15	0_0	0.10	0.17	0_1	0.09	0.18
SD520	0_9	0.00	0.03	0_2	0.10	0.18	0_1	0.04	0.14
SD550	0_4	0.08	0.16	0_2	0.09	0.18	0.05	0.14	0.17
SD5100	0_8	0.01	0.03	0_1	0.08	0.16	0_2	0.03\4	0.11
MD320	0_6	0.04	0.10	0_1	0.06	0.17	0.04	0.11	0.21
MD350	0_5	0.04	0.13	0_3	0.02	0.07	0.03	0.08	0.22
MD3100	0_6	0.04	0.12	0_4	0.06	0.11	0.03	0.08	0.18
MD520	0_6	0.03	0.09	0_6	0.01	0.06	0.09	0.18	0.27
MD550	0_7	0.01	0.06	0_5	0.03	0.07	0.03	0.07	0.18
MD5100	0_6	0.02	0.06	0_6	0.01	0.04	0_1	0.04	0.09

ANOVA test for significant difference between all methods are given in Table 6.

Base on Table 5, NGA and improved IACO are two most powerful method to find final solution. Table 7 and 8 shows detail comparison between these methods.

Table 7 demonstrated that NGA outperformed IACO in 73% of all classes and all of the differences are statistically significant. Also, ACO outperformed NGA in 27% of all classes and in all cases, differences are statistically significant.

Table 6: ANOVA test for all methods

	Some of Square(SS)	Degree of freedom(df)	Mean Square(MS)	VR	F
Between groups (or "Factor")	68999512.3	3	22999837.4	4.8	2.8
Within groups (or "Error")	267070448.9	56	4769115.2		
Total	336069961.2	59			

Table 7: t test to evaluate significance of differences

Instance	**Average** ($\bar{X}$)		**Standard Deviation** (S)		T	**Significance**
	NGA	IACO	NGA	IACO		
SD120	861.12	915.45	3.52	4.64	29.49	Yes
SD150	962.15	1111.31	6.94	12.26	33.48	Yes
SD1100	3058.82	3425.64	16.39	15.39	51.59	Yes
SD320	5877.92	6513.91	20.69	22.05	66.52	Yes
SD350	11696.41	12764.26	30.78	34.89	72.58	Yes
SD3100	27002.15	27099.83	49.37	51.38	4.34	Yes
SD520	40909.38	40460.98	62.18	61.15	-16.26	Yes
SD550	106180.59	110482.91	67.25	68.48	141.75	Yes
SD5100	198183.35	195964.18	64.84	67.39	-75.04	Yes
MD320	5804.31	5962.83	22.00	26.74	14.48	Yes
MD350	11404.94	12631.54	30.21	26.38	96.71	Yes
MD3100	27378.64	27492.01	49.71	47.45	5.22	Yes
MD520	41184.98	44304.38	55.66	58.92	121.71	Yes
MD550	92254.74	90701.83	72.98	57.66	-52.80	Yes
MD5100	204431.64	201865.34	71.42	88.58	-71.32	Yes
Each Ins contains 10 independent test						

Table 8: Tukey test results for NGA and IACO

	$\bar{x}_{ACO} - \bar{x}_{LA}$	**Critical q** $q(\alpha, r, df_w)$	95% Conf Interval for $\mu_{ACO} - \mu_{LA}$		**Significant at 0.05?**
			Min	Max	
$HGA - LA$	300.35	2.75	-180.24	180.11	Yes

5 Conclusion

In this research, we introduced one population-based meta-heuristic method. The presented method is based on the lifestyle of the Lion and the genetic algorithm's structure; therefore we called it as New Genetic Algorithm (NGA). The initial population of solutions distribute between some groups (pride) and the best solution within each group is called commander. Each child in each group (earned by mutation or crossover) challenges all commanders to substitute with them. After a finite number of iterations, the best commander is the final solution. For the verification test, we consider two main N-P hard as flow-shop scheduling and VRP. We recalled two powerful methods from the literature as hybrid genetic algorithm and ant colony optimization. Based on the comparison study, NGA works very competitive to solve N-P hard problems.

References

[1] J. Alegre, M. Laguna and J. Pacheco, *Optimizing the periodic pick-up of raw materials for a manufacturer of auto parts*, Eur. J. Oper. Res. **179** (2007), no. 3, 736–746.

[2] I. Benkalai, D. Rebaine, C. Gagne and P. Baptiste, *Improving the migrating birds optimization metaheuristic for the permutation flow shop with sequence-dependent set-up times*, Int. J. Prod. Res. **55** (2017), no. 20, 6145–6157.

[3] C. Blum and R. Andrea, *Metaheuristics in combinatorial optimization, overview and conceptual comparison*, ACM Comput. Surv. **35** (2003), no. 3, 268-–308.

[4] H.G. Campbell, R.A. Dudek and M.L. Smith, *A heuristic algorithm for the n job, m machine sequencing problem*, Manag. Sci. **16** (1970), no. 10, 630—637.

[5] I. Chao, B.L. Golden and E. Wasil, *A new heuristic for the multi-depot vehicle routing problem that improves upon best known solutions*, Amer. J. Math. Manag. Sci. **13** (2007), 371—406.

[6] J.F. Cordeau, M. Gendreau and G. Laporte, *A tabu search heuristic for periodic and multi-depot vehicle routing problems*, Networks **30** (1997), 105-–119.

[7] J.F. Cordeau, G. Laporte and A. Mercier, *A unified tabu search heuristic for vehicle routing problems with time windows*, J. Oper. Res. Soc. **52** (2001), no. 8, 928–936.

[8] B. Crevier, J. Cordeau and G. Laporte, *The multi-depot vehicle routing problem with inter-depot routes*, Eur. J. Oper. Res. **176** (2007), 756–773.

[9] Q. Cui, W.U. Xiuli and Y.U. Jianjun, *Improved genetic algorithm variable neighborhood search for solving hybrid flow shop scheduling problem*, Comput. Integr. Manuf. Syst. **23** (2017), 1917-–1927.

[10] P.R.D.O. Da Costa, S. Mauceri, P. Carroll and F. Pallonetto, *A genetic algorithm for a green vehicle routing problem*, Electron. Notes Discrete Math. **64** (2018), 65–74.

[11] G.B. Dantzig and J.H. Ramser, *The truck dispatching problem*, Manag. Sci. **6** (1959), no. 1, 80–91.

[12] L.M.A. Drummond, L.S. Ochi and D.S. Vianna, *An asynchronous parallel metaheuristics for the period routing problem*, Future Generation Comput. Syst. **17** (2001), no. 4, 379–386.

[13] T. El-Ghazali, *Meta-heuristics, from design to implementation*, John Wiley & Sons, 2009.

[14] M.R. Garey, D.S. Johnson and R. Sethi, *The complexity of flow-shop and job-shop scheduling*, Math. Operat. Res. **1** (1976), 117—129.

[15] I.D. Giosa, I.L. Tansini and I.O. Viera, *New assignment algorithms for the multi-depot vehicle routing problem*, J. Oper. Res. Soc. **53** (2002), 977-–984.

[16] V.C. Hemmelmayr, K.F. Doerner and R.F. Hartl, *A variable neighborhood search heuristic for periodic routing problems*, Eur. J. Oper. Res. **195** (2009), no. 3, 791–802.

[17] W. Ho, T.S. Ho, P. Ji and C.W. Lau, *A hybrid genetic algorithm for the multi-depot vehicle routing problem*, Engin. Appl. Artif. Intell. **21** (2008), 548–557.

[18] W.L. Jin and M. Price, *New meta-heuristic for dynamic scheduling in permutation flowshop with new order arrival*, Int. J. Adv. Manuf. Technol. **98** (2018), no. 5-8, 1817–1830.

[19] S.M. Johnson, *Optimal two and three-stage production schedules with setup times included*, Naval Res. Logist. Quart. **1** (1954), 61—68.

[20] A. Lim and W. Zhu, *A fast and effective insertion algorithm for MDVRP with fixed distribution of vehicles and a new simulated annealing approach*, The 19th Int. Conf. Ind. Engin. Other Appl. Appl. Intell. Syst. (IEA/AIE'06), 2006, p. 282–291.

[21] M. Mirabi, *A novel hybrid genetic algorithm to solve the sequence dependent permutation flow-shop scheduling problem*, Int. J. Adv. Manuf. Technol. **71** (2014), no. 1, 429–437.

[22] M. Mirabi, *A novel hybrid genetic algorithm for the multidepot periodic vehicle routing problem*, AI EDAM. **29** (2015), no. 1, 45–54.

[23] M. Nawaz, E.E. Enscore and I. Ham, *A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem*, Omega **11** (1983), no. 1, 91—95.

[24] I.H. Osman and G. Laporte, *Metaheuristics: A bibliography*, Ann. Oper. Res. **63** (1996), 513—623.

[25] I.H. Osman and C.N. Potts, *Simulated annealing for permutation flow-shop scheduling*, Omega **17** (1989), no. 6, 551-–557.

[26] Q. Pan, L. Gao, L. Wang, J. Liang and X. Li, *Effective heuristics and meta-heuristics to minimize total flow-time for the distributed permutation flowshop problem*, Expert Syst. Appl. **124** (2019), no. 15, 309–324.

[27] D. Pisinger and S. Ropke, *A general heuristic for the vehicle routing problems*, Comput. Oper. Res. **34** (2007), no. 8, 2403–2435.

[28] M. Reed, A. Yiannakou and R. Evering, *An ant colony algorithm for the multi-compartment vehicle routing problem*, Appl. Soft Comput. **15** (2014), 169-–176.

[29] C.R. Reeves, *A genetic algorithm for flow-shop sequencing*, Comput. Oper. Res. **22** (1995), 5—13.

[30] J. Renaud, G. Laporte and F.F. Boctor, *A tabu search heuristic for the multi-depot vehicle routing problem*, Comput. Oper. Res. **23** (1996), 229-–235.

[31] R. Ruiz, C. Maroto and J. Alcaraz, *Solving the flow-shop scheduling problem with sequence dependent setup times using advanced meta-heuristics*, Eur. J. Oper. Res. **165** (2005), 34—54.

[32] R. Ruiz and T. Stutzle, *An iterated greedy heuristic for the sequence dependent setup times flow-shop with make-span and weighted tardiness objectives*, Eur. J. Oper. Res. **87** (2008), 1143–1159.

[33] P. Stodola, *Using meta-heuristics on the multi-depot vehicle routing problem with modified optimization criterion*, Algorithms **11** (2018), no. 5.

[34] J.U. Sun and H. Hwang, *Scheduling problem in a two machine flow line with the N-step prior-job-dependent set-up times*, Int. J. Syst. Sci. **32** (2001), 375-–385.

[35] S. Thangiah and S. Salhi, *Genetic clustering: an adaptive heuristic for the multi depot vehicle routing problem*, Appl. Artif. Intell. **15** (2001), no. 4, 361–383.

[36] M. Widmer and A. Hertz, *A new heuristic method for the flow shop sequencing problem*, European J. Operat. Res. **41** (1989), 186-–193.

[37] X. Zhang, J. Tong and M. Yunyong, *An effective hybrid ant colony optimization for permutation flow-shop scheduling*, Open Autom. Cont. Syst. J. **6** (2014), 62–68.