# Adopting Jaya Algorithm for Team Formation Problem

Md. Abdul Kader
Faculty of Computing
Universiti Malaysia Pahang
Pahang, Malaysia
kdr2k10@gmail.com

Kamal Z. Zamli
Faculty of Computing
Universiti Malaysia Pahang
Pahang, Malaysia
kamalz@ump.edu.my

## ABSTRACT
This paper presents a simple and mighty metaheuristic algorithm, Jaya, which is applied to solve the team formation (TF) problem and it is a very fundamental problem in many databases and expert collaboration networks or web applications. The Jaya does not need any distinctive parameters that require comprehensive tuning, which is usually troublesome and inefficient. Among several optimization methods, Jaya is chosen for TFP because of its simplicity and it always avoids the worst solutions and moving towards the global best solution. This victorious nature makes Jaya Algorithm more powerful and significant as compared to any other contemporary optimization algorithms. To evaluate the efficiency of the Jaya Algorithm (JA) against another metaheuristic algorithm, Sine-Cosine Algorithm (SCA), both algorithms are tested and assessed for the TF problem solution using an ACM dataset containing experts and their skills. The experimental results validate the improved performance of the optimization solutions and the potential of JA with fast convergence for solving TF problems which are better than SCA.

## CCS Concepts
**• Mathematics of computing → Mathematical analysis → Mathematical optimization → Non-parametric optimization**

## Keywords
Team formation problem; Jaya optimization algorithm; Sine-Cosine algorithm; Metaheuristic algorithm; Application of Jaya algorithm.

## 1. INTRODUCTION
Recently, it is observed that dealing with different tasks through collaboration with experts is increasing massively in any network. Finding the experts and creating the best groups or teams for specific jobs with required skills is always a challenging task. This challenge leads to the research of the team formation (TF) problem in many real-life applications [1-8]. The TF problem goal is to find the best group of experts who have the desired skills and cost-effective communication with other experts [8]. Although technological improvement eliminates the geographical and local constraints, there are several methods independently proposed for calculating communication costs between experts in literature [1-5, 8].


*ICSCA 2020,* February 18–21, 2020, Langkawi, Malaysia

ACM ISBN 978-1-4503-7665-5/20/02…$15.00
https://doi.org/10.1145/3384544.3384593

Researchers investigated and proposed many strategies on how to find the best team adequately [5, 7, 9-14]. Among them, metaheuristics often work much better in practice. These algorithms are popular because of their simplicity, easy implementation in any programming language, and solution diversity. There are two principal phases in metaheuristics termed as exploitation and exploration. Exploitation uses neighbor information obtained from local search and tries to find the local best solution, while the exploration phase tends to explore different feasible regions in the whole search area and tries to find the global best solution [15]. The challenge is to balance these two phases of metaheuristics, which affects the performance of these algorithms.

In literature, metaheuristics algorithms are classified into two broad categories, such as single solution based metaheuristic algorithms and multi-solution based or population-based metaheuristic algorithms [16]. Population-based metaheuristic algorithms are further classified into two broad categories. These are evolutionary-based and swarm-based metaheuristic algorithms [17]. However, both types are probabilistic and require standard controlling parameters. Besides, some of them need their algorithm-specific control parameters. The improper tuning of these parameters either provides the local optimal solutions or increases the computational cost [18].

Previous research reported many optimization techniques to solve TF problems. In this paper, a well-known distinct parameter-free metaheuristic algorithm, Jaya, which is proposed by R. Venkata Rao in 2016 [18] is chosen for the TF problem solution because of its simplicity and robustness and researchers used it in many applications like [19-21]. To the best of understanding of the writer, there is no previous study used the most straightforward Jaya algorithm in this TF problem area and this paper aims to take this opportunity to find out the JA potential in the TF problem solution. Furthermore, to make the experiment fair, the efficiency of JA is compared with the Sine-Cosine Algorithm (SCA) [22] for the TF problem solution. SCA is another new population-based metaheuristic and it is useful in solving real problems [23].

This paper is structured accordingly as follows. Section 2, describes an overview of the Jaya algorithm. Section 3 delivers the technique of using Jaya in the TF problem. Section 4 describes the performance comparison of JA and SCA from the results of the experiment and the description of the work. Finally, Section 5 concludes the paper.

## 2. OVERVIEW OF JAYA ALGORITHM
Jaya is a meta-heuristic algorithm that is easier, more efficient, and more powerful algorithm for finding the global best solution. It has been applied to many benchmark functions for constrained and unconstrained problems successfully. Additionally, Jaya is like TLBO which has two phases (teacher and learner) [24], but JA does not require the learner phase [18]. Jaya works by establishing the solution to problems through avoiding the worst solutions and moving towards the best optimal solution. Although

it is algorithm-specific parameter-free, its performance depends on only a few control parameters, which are common to many optimization algorithms like population size, number of generations, and number of design variables. The working principle of the Jaya algorithm is described below.

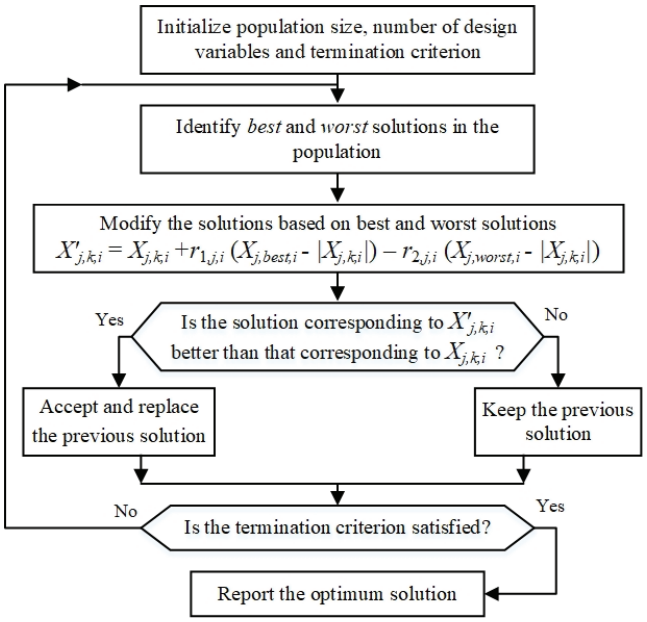


**Figure 1. Flowchart of the Jaya algorithm.**

All the terminologies assumed and used in JA are shown in the following Table 1.

**Table 1. Terminologies used in Jaya**

| Terms | Explanation |
|---|---|
| Objective Function | $f(x)$ |
| Population Size | $k=1,2,\ldots,n$ |
| Design Variables | $j=1,2,\ldots,m$ |
| *best* Candidate | $f(x)_{best}$ |
| *worst* Candidate | $f(x)_{worst}$ |
| $X_{j,k,i}$ | $j^{th}$ design variable for the $k^{th}$ candidate at $i^{th}$ iteration |
| $X'_{j,k,i}$ | Modification of $X_{j,k,i}$ |
| $X_{j,best,i}$ | $j$ variable for the best |
| $X_{j,worst,i}$ | $j$ variable for the worst |
| $r_{1,j,i}$ | random number in the range of [0, 1] |
| $r_{2,j,i}$ | random number in the range of [0, 1] |

The Jaya algorithm begins by adjusting its basic parameters. These are termination criteria (here the maximum number of iterations are considered as the termination condition), population size (number of candidate solutions), and number of design variables. In the second and third steps, the best and worst solution is identified from the population and modify the solutions according to Equation 1, which is for the $j^{th}$ variable at $i^{th}$ iteration.

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,best,i} - |X_{j,k,i}|) - r_{2,j,i}(X_{j,worst,i} - |X_{j,k,i}|) \quad (1)$$

In the fourth step shown in Figure 1, it is seen that if the updated solution is better than the previous solution, then it accepts and replaces the previous solution otherwise keeps the previous solution. In the final step, if the termination condition is satisfied, then it reports the optimum solution; otherwise, get back to the second step of this algorithm.

By analyzing Equation 1, it can be clear that the search avoids the worst solutions, i.e., moving away from the worst solution and keep moving towards the best solution, i.e., closer to the success. Additionally, finding the best solution or its convergence rate is faster than the others.

## 3. TEAM FORMATION PROBLEM USING JAYA ALGORITHM

Now we formally define the TF problem that we address in this paper. Let $E$ be the set of $m$ experts or design variables (i.e. $j$ = 1, 2,....,$m$). $S$ is the set of all unique skills for all the experts in $E$. $C$ is the cost set between each pair of experts in $E$. $n$ is the number of candidate solutions or population size. $T$ is the set of $n$ candidate solutions (i.e. $k$ = 1,2,…,$n$). $M_{itr}$ is the maximum number of iterations used for termination criteria. $f(T_{j,k,i})$ is the objective function or cost function where $i$ = 1,2,…, $M_{itr,}$ and it returns the cost of $j^{th}$ design variable for the $k^{th}$ candidate at the $i^{th}$ iteration. Algorithm 1 shows how Jaya can be used in the TF problem solution to get the best optimum solution.

**Algorithm 1: TF Problem Using Jaya Algorithm**

**Input:** The population $T_k \leftarrow [k=1,2,\ldots,n]$
**Output:** The best obtained Team
1 : **procedure** TFJA
2 : Initialize the parameters $m$, $n$, $M_{itr}$
3 : Calculate fitness $f(T)$
4 : while($i < M_{itr}$) do
5 : for $k \leftarrow 1$ to $n$ do
6 : Identify $f(T)_{best}$ and $f(T)_{worst}$
7 : for $j \leftarrow 1$ to $m$ do
8 : $r_{1,j,i} \leftarrow$ rand() and $r_{2,j,i} \leftarrow$ rand()
9 : Update the solutions using the following equation
10: $T'_{j,k,i} = T_{j,k,i} + r_{1,j,i}(T_{j,best,i} - |T_{j,k,i}|) - r_{2,j,i}(T_{j,worst,i} - |T_{j,k,i}|)$
11: Calculate fitness $f(T)$
12: if ($f(T'_{j,k,i}) < f(T_{j,k,i})$) then
13: Update the earlier solution
14: end if
15: end for
16: end for
17: $i \leftarrow i+1$
18: end while
19: return best obtained Team
20: end procedure

In Algorithm 1, the required parameters are initialized, and the pre fitness of all candidates is calculated before entering the main loop. The main loop will run $M_{itr}$ times at best. After identifying the $f(T)_{best}$ and $f(T)_{worst}$, all solutions are calculated using Equation 1 and updated if the best solution is found out (line no. 7 to line no. 15 of Algorithm 1) and this process will continue for each population (line no. 5 to line no. 16 of Algorithm 1). The same process repeated (line no. 4 to line no. 18 of Algorithm 1) until the maximum fitness covered or reaching the maximum iteration.

## 4. EXPERIMENTAL RESULTS AND DISCUSSION

JA (Algorithm 1) and SCA were coded in Java, run in windows 10 using CPU Intel core i7 2.20 GHz speed, and 8GB RAM. Although JA can be tested for any number of skills between 1 and the total number of unique skills that exist in the corresponding dataset, this algorithm is tested for five sets of skills which include 10, 20, 30, 40, 50 skills. For each skill set, both JA and SCA run by ten times for the same set of population/candidate solutions and taken the average of all results to make the experiment fair. To test the performance of Jaya and Sine-Cosine Algorithms, a middle size formatted and cleaned dataset, ACM dataset (https://github.com/MAK660/Dataset/blob/master/Experts_Skills_Dataset.txt), which contains a total of 3702 experts and 5197 unique skills is used in this experiment. As we aim to measure the efficiency of the Jaya algorithm, Table 2 to Table 6 shows the experimental results of this study. The average of best, max, mean (for all populations) values of team size and team cost calculated and shown at the bottom of each table (Table 2 to Table 6) and they are presented graphically in Figure 2 and Figure 3. Although the performance of optimization depends on the number of required skills to covered by the number of experts, it is observed from Figure 2 and Figure 3 that the value of team size and team cost are proportional to the number of required skills to find for the optimization of TF problems.

An experimental comparison of JA and SCA for each skillset is shown in Table 7. The average (10 runs) of best and mean values of team size and team cost and their average running time(ms) for the mentioned test skills are stated for comparison. Figure 4 shows the comparison of best and mean team sizes (TS) for the different sets of test skills between JA and SCA. Figure 5 show the comparison of the best and mean team cost (TC) for the different sets of test skills between JA and SCA. The average running time (ms) comparison is shown in Figure 6. This experimental comparison shows a noticeable improvement that all the average results (team size, team cost and running time) are close to the best outcome, and JA always performs better than SCA for the optimization of TF problems. Moreover, a graphical analysis of Team size (TS) vs. Team Cost (TC) for best and mean value is shown in Figure 7 where the team size vs. team cost curve of JA is always better (less) than the team size vs. team cost curve of SCA in both cases.

## 5. CONCLUSION

In this paper, we presented a brief knowledge of the team formation problem and reviewed the working principle of the Jaya algorithm. The reasons for choosing Jaya are mentioned when handling optimization problems like the TF problem. From Table 2-6, it is observed that average results of team size (best, max, mean) and team cost (best, max, mean) are close to the best solution, which is acceptable for the TF problem. To make the experiment fair and to analyze the efficiency of JA, SCA is used in this study. From the experimental comparison of JA and SCA shown in Table 7, Jaya proves its computation capability of faster convergence than SCA in the TF problem solution. The authors have strongly presented that this work achieves what it was looking for effectively.


## 6. ACKNOWLEDGMENT

This research is funded by RDU Grant No. UIC191202: The Development of T-Way Test Generation Tool for Combinatorial Testing from Universiti Malaysia Pahang.


**Table 2. Experimental results for ten skills set.**

| Number of Skills: 10 | | | | | | |
|---|---|---|---|---|---|---|
| | Team Size | | | Team Cost | | |
| Run No. | Best | Max | Mean | Best | Max | Mean |
| 1 | 3 | 5 | 4.00 | 3.00 | 09.47 | 5.91 |
| 2 | 3 | 7 | 4.50 | 3.00 | 19.10 | 8.07 |
| 3 | 3 | 6 | 4.90 | 3.00 | 14.37 | 9.22 |
| 4 | 3 | 6 | 4.00 | 3.00 | 13.57 | 6.05 |
| 5 | 3 | 5 | 4.00 | 3.00 | 09.63 | 5.96 |
| 6 | 3 | 6 | 4.60 | 3.00 | 13.27 | 7.72 |
| 7 | 3 | 5 | 4.00 | 3.00 | 09.57 | 5.95 |
| 8 | 4 | 5 | 4.50 | 5.80 | 09.70 | 7.48 |
| 9 | 3 | 6 | 4.30 | 3.00 | 14.06 | 7.24 |
| 10 | 3 | 6 | 3.90 | 3.00 | 13.48 | 5.76 |
| Average: | 3.10 | 5.70 | 4.27 | 3.28 | 12.62 | 6.93 |

**Table 3. Experimental results for twenty skills set.**

| Number of Skills: 20 | | | | | | |
|---|---|---|---|---|---|---|
| | Team Size | | | Team Cost | | |
| Run No. | Best | Max | Mean | Best | Max | Mean |
| 1 | 9 | 17 | 13.70 | 34.40 | 126.38 | 84.89 |
| 2 | 11 | 19 | 14.30 | 51.47 | 162.93 | 93.39 |
| 3 | 12 | 17 | 14.30 | 63.26 | 127.87 | 92.09 |
| 4 | 11 | 18 | 14.60 | 53.19 | 144.46 | 96.90 |
| 5 | 8 | 18 | 13.40 | 27.58 | 144.87 | 84.31 |
| 6 | 11 | 19 | 15.20 | 53.68 | 160.71 | 104.96 |
| 7 | 8 | 17 | 12.90 | 27.76 | 123.47 | 76.76 |
| 8 | 13 | 21 | 16.00 | 74.53 | 201.17 | 117.31 |
| 9 | 8 | 17 | 13.20 | 27.02 | 129.22 | 78.86 |
| 10 | 8 | 18 | 13.30 | 27.72 | 142.36 | 81.19 |
| Average: | 9.90 | 18.10 | 14.09 | 44.06 | 146.34 | 91.07 |

**Table 4. Experimental results for thirty skills set.**

| Number of Skills: 30 | | | | | | |
|---|---|---|---|---|---|---|
| | Team Size | | | Team Cost | | |
| Run No. | Best | Max | Mean | Best | Max | Mean |
| 1 | 17 | 32 | 24.80 | 131.73 | 477.11 | 293.77 |
| 2 | 16 | 26 | 21.70 | 116.71 | 314.63 | 222.13 |
| 3 | 18 | 33 | 24.50 | 150.26 | 509.70 | 287.80 |
| 4 | 18 | 29 | 23.00 | 145.68 | 388.54 | 250.36 |
| 5 | 17 | 30 | 25.70 | 132.36 | 423.08 | 313.24 |
| 6 | 22 | 27 | 24.40 | 221.84 | 338.35 | 275.87 |
| 7 | 20 | 31 | 25.50 | 185.42 | 443.79 | 305.36 |
| 8 | 17 | 31 | 24.10 | 134.34 | 448.98 | 278.32 |
| 9 | 17 | 30 | 23.00 | 132.12 | 417.83 | 252.34 |
| 10 | 20 | 29 | 24.20 | 184.00 | 389.72 | 274.56 |
| Average: | 18.20 | 29.80 | 24.09 | 153.45 | 415.17 | 275.38 |

**Table 5. Experimental results for forty skills set.**

| Number of Skills: 40 | | | | | | |
|---|---|---|---|---|---|---|
| | Team Size | | | Team Cost | | |
| Run No. | Best | Max | Mean | Best | Max | Mean |
| 1 | 36 | 57 | 43.60 | 610.97 | 1543.96 | 917.37 |
| 2 | 29 | 51 | 40.80 | 397.12 | 1244.37 | 820.39 |
| 3 | 30 | 51 | 42.70 | 424.48 | 1238.25 | 880.23 |
| 4 | 31 | 50 | 44.00 | 450.86 | 1185.48 | 932.41 |
| 5 | 36 | 50 | 42.00 | 614.91 | 1191.22 | 844.63 |
| 6 | 27 | 49 | 40.30 | 344.17 | 1148.26 | 798.02 |
| 7 | 32 | 49 | 40.70 | 482.69 | 1144.75 | 795.58 |
| 8 | 32 | 50 | 39.70 | 482.02 | 1189.76 | 765.29 |
| 9 | 28 | 52 | 44.20 | 368.37 | 1284.70 | 949.86 |
| 10 | 33 | 50 | 40.70 | 514.76 | 1175.92 | 802.67 |
| Average: | 31.40 | 50.90 | 41.87 | 469.03 | 1234.67 | 850.65 |

**Table 6. Experimental results for fifty skills set.**

| Number of Skills: 50 | | | | | | |
|---|---|---|---|---|---|---|
| | Team Size | | | Team Cost | | |
| Run No. | Best | Max | Mean | Best | Max | Mean |
| 1 | 58 | 82 | 69.40 | 1619.98 | 3243.82 | 2356.47 |
| 2 | 56 | 79 | 70.60 | 1507.68 | 3013.40 | 2429.45 |
| 3 | 42 | 76 | 62.00 | 0839.58 | 2786.84 | 1886.74 |
| 4 | 35 | 77 | 64.30 | 0587.31 | 2858.46 | 2057.60 |
| 5 | 54 | 80 | 63.80 | 1400.70 | 3072.34 | 1998.80 |
| 6 | 54 | 87 | 68.50 | 1401.32 | 3656.84 | 2330.99 |
| 7 | 54 | 86 | 66.70 | 1400.93 | 3557.04 | 2199.50 |
| 8 | 54 | 80 | 68.00 | 1403.77 | 3086.72 | 2256.34 |
| 9 | 49 | 76 | 61.70 | 1156.39 | 2790.19 | 1872.08 |
| 10 | 50 | 83 | 67.80 | 1201.66 | 3332.85 | 2256.43 |
| Average: | 50.60 | 80.60 | 66.28 | 1251.93 | 3139.85 | 2164.44 |

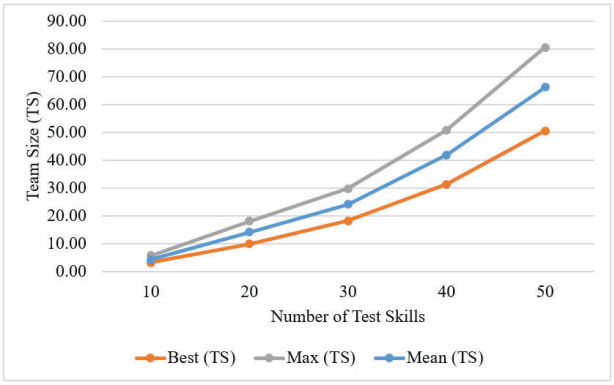


**Figure 2. The average value of best, max and mean team size (TS) for the different sets of test skills using JA.**

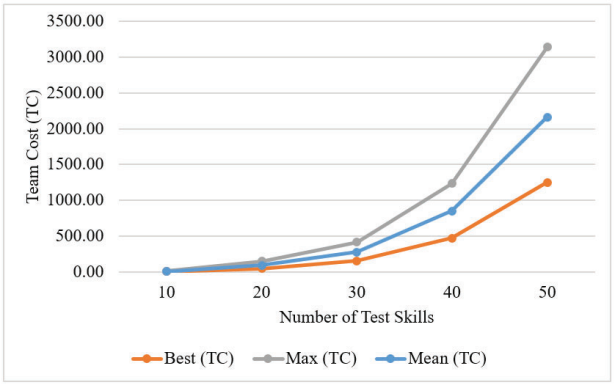


**Figure 3. The average value of best, max and mean team cost (TC) for the different sets of test skills using JA.**

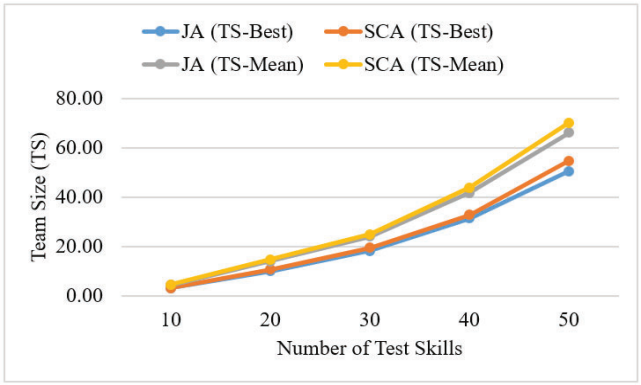


**Figure 4. Comparison of best and mean team size (TS) for the different sets of test skills between JA and SCA.**

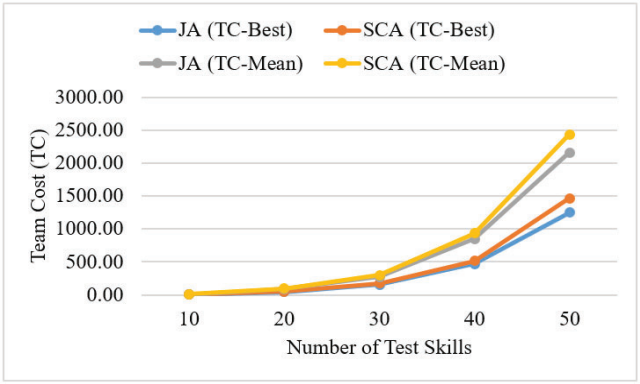


**Figure 5. Comparison of best and mean team cost (TC) for the different sets of test skills between JA and SCA.**

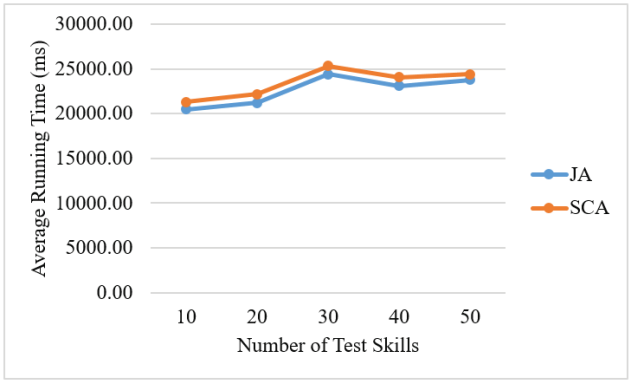


**Figure 6. Average running time comparison for the different sets of test skills between JA and SCA.**

**Table 7. Performance comparison of JA and SCA**

| Test Skills | Team Size | | | | Team Cost | | | | Average Running Time (ms) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Best | | Mean | | Best | | Mean | | | |
| | JA | SCA | JA | SCA | JA | SCA | JA | SCA | JA | SCA |
| 10 | 3.10 | 3.10 | 4.27 | 4.50 | 3.28 | 3.28 | 6.93 | 7.83 | 20487.90 | 21332.50 |
| 20 | 9.90 | 10.60 | 14.09 | 14.64 | 44.06 | 49.89 | 91.07 | 98.95 | 21237.30 | 22191.40 |
| 30 | 18.20 | 19.40 | 24.09 | 24.95 | 153.45 | 175.16 | 275.38 | 297.14 | 24407.00 | 25342.90 |
| 40 | 31.40 | 32.80 | 41.87 | 43.94 | 469.03 | 513.51 | 850.65 | 934.65 | 23107.10 | 24076.00 |
| 50 | 50.60 | 54.90 | 66.28 | 70.34 | 1251.93 | 1468.00 | 2164.44 | 2440.65 | 23789.20 | 24419.50 |

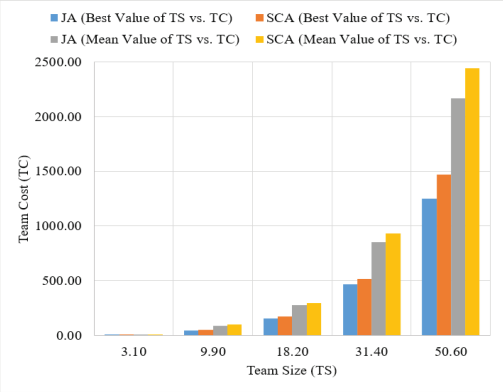


**Figure 7. Team Size (TS) vs. Team Cost (TC) analysis of JA and SCA.**

# 7. REFERENCES


[1] T. Lappas, K. Liu, and E. Terzi, "Finding a team of experts in social networks," presented at the Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, Paris, France, 2009.

[2] M. Kargar and A. An, "Discovering top-k teams of experts with/without a leader in social networks," presented at the Proceedings of the 20th ACM international conference on Information and knowledge management, Glasgow, Scotland, UK, 2011.

[3] M. Kargar, A. An, and M. Zihayat, "Efficient bi-objective team formation in social networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2012: Springer, pp. 483-498.

[4] A. Majumder, S. Datta, and K. V. M. Naidu, "Capacitated team formation problem on social networks," presented at the Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, Beijing, China, 2012.

[5] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi, *Online Team Formation in Social Networks*. 2012.

[6] M. Kargar, M. Zihayat, and A. An, "Finding Affordable and Collaborative Teams from a Network of Experts," 2013, pp. 587-595.

[7] S. Rangapuram, T. Bühler, and M. Hein, *Towards Realistic Team Formation in Social Networks based on Densest Subgraphs*. 2013, pp. 1077-1088.

[8] X. Wang, Z. Zhao, and W. Ng, "USTF: A Unified System of Team Formation," *IEEE Transactions on Big Data,* vol. 2, no. 1, pp. 70-84, 2016, doi: 10.1109/TBDATA.2016.2546303.

[9] M. Fathian, M. Saei-Shahi, and A. Makui, "A New Optimization Model for Reliable Team Formation Problem Considering Experts' Collaboration Network," *IEEE Transactions on Engineering Management,* vol. 64, no. 4, pp. 586-593, 2017, doi: 10.1109/TEM.2017.2715825.

[10] J. Basiri, F. Taghiyareh, and A. Ghorbani, "Collaborative team formation using brain drain optimization: a practical and effective solution," *World Wide Web,* 02/18 2017, doi: 10.1007/s11280-017-0440-6.

[11] L. Li, H. Tong, N. Cao, K. Ehrlich, Y.-R. Lin, and N. Buchler, *TEAMOPT: Interactive Team Optimization in Big Networks*. 2016, pp. 2485-2487.

[12] A. Farasat and A. G. Nikolaev, "Social Structure Optimization in Team Formation," *Computers & Operations Research,* vol. 74, 04/01 2016, doi: 10.1016/j.cor.2016.04.028.

[13] V. Baghel and S. Bhavani, *Multiple Team Formation using an Evolutionary Approach*. 2019.

[14] A. A. Alsewari and K. Z. Zamli, "Interaction test data generation using harmony search algorithm," in *2011 IEEE Symposium on Industrial Electronics and Applications*, 2011: IEEE, pp. 559-564.

[15] X.-S. Yang, "Nature-Inspired Mateheuristic Algorithms: Success and New Challenges," *Journal of Computer Engineering and Information Technology,* vol. 01, 11/28 2012, doi: 10.4172/2324-9307.1000e101.

[16] T. F. Ghanem, W. S. Elkilani, and H. M. Abdul-Kader, "A hybrid approach for efficient anomaly detection using metaheuristic methods," *Journal of advanced research,* vol. 6, no. 4, pp. 609-619, 2015.

[17] H. M. Pandey, "Jaya a novel optimization algorithm: What, how and why?," in *2016 6th International Conference - Cloud System and Big Data Engineering (Confluence)*, 14-15 Jan. 2016 2016, pp. 728-730, doi: 10.1109/CONFLUENCE.2016.7508215.

[18] R. Venkata Rao, "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems," *International Journal of Industrial Engineering Computations,* vol. 7, pp. 19-34, 01/01 2016, doi: 10.5267/j.ijiec.2015.8.004.

[19] A. Alomoush, A. Alsewari, H. Alamri, and K. Zamli, "Solving 0/1 Knapsack Problem Using Hybrid HS and Jaya Algorithms," *Advanced Science Letters,* vol. 24, pp. 7486-7489, 10/01 2018, doi: 10.1166/asl.2018.12964.

[20] A. B. Nasser, F. Hujainah, A. A. Al-Sewari, and K. Z. Zamli, "An Improved Jaya Algorithm-Based Strategy for T-Way Test Suite Generation," Cham, 2020: Springer International Publishing, in Emerging Trends in Intelligent Computing and Informatics, pp. 352-361.

[21] K. Zamli, A. Alsewari, and B. Ahmed, "Multi-Start Jaya Algorithm for Software Module Clustering Problem," *Azerbaijan Journal of High Performance Computing,* vol. 1, pp. 87-112, 08/12 2018, doi: 10.32010/26166127.2018.1.1.87.112.

[22] S. Mirjalili, "SCA: a sine cosine algorithm for solving optimization problems," *Knowledge-Based Systems,* vol. 96, pp. 120-133, 2016.

[23] K. Z. Zamli, F. Din, B. S. Ahmed, and M. Bures, "A hybrid Q-learning sine-cosine-based strategy for addressing the combinatorial test suite minimization problem," *PloS one,* vol. 13, no. 5, p. e0195675, 2018.

[24] R. V. Rao, V. Savsani, and J. Balic, "Teaching–learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems," *Engineering Optimization,* vol. 44, no. 12, pp. 1447-1462, 2012.