10.1109/TETC.2018.2812927

A New Class Topper Optimization Algorithm with an Application to Data Clustering

Pranesh Das, Dushmanta Kumar Das, *Member, IEEE* and Shouvik Dey

Abstract

In this paper, a new Class Topper Optimization (CTO) algorithm is proposed. The optimization algorithm is inspired from the learning intelligence of students in a class. The algorithm is population based search algorithm. In this approach, solution is converging towards the best solution. This may lead to a global best solution. To verify the performance of the algorithm, a clustering problem is considered. Five standard data sets are considered for real time validation. The analysis shows that the proposed algorithm performs very well compared to various well known existing heuristic or meta-heuristic optimization algorithms.

Index Terms—Data clustering, Optimization algorithm, Learning intelligence, Data analysis and Nature inspired optimization

1 Introduction

Many nature inspired meta-heuristic optimization algorithms are developed for its wide applications in engineering, economics, biology, etc. But, researchers always welcome new optimization algorithms. This gives a boosting energy to think the activities happening around us. In this paper, a new class topper optimization (CTO) algorithm is proposed. The proposed CTO algorithm is based on the intelligence behavior of students in a particular class of a school. In a particular class of students, a competitive behavior is being observed to become the topper of that class. This competitiveness among the students make them better in their performance in the class. This intelligence behavior of the students can be modeled to solve many real life complex optimization problems. In this paper, a complex data analysis task, i.e. cluster analysis is considered to check the effectiveness of the proposed algorithm.

Many new heuristic or meta-heuristic optimization algorithms are introduced by many researchers to solve various real life problems. In [1], a new bionic optimization algorithm named Magnetotactic Bacteria Moment Migration Algorithm (MBMMA) is proposed, in which the moments of relative good solutions can migrate each other to enhance the diversity of the MBMMA. A nature-inspired rain-fall optimization algorithm (RFO) based on behavior of raindrops, for solving of real-valued numerical optimization problems is proposed in [2]. It is effective in searching and finding an optimum solution from a large search domain within an acceptable CPU time. In-order to find the optimal path of data transmission in the wireless sensor networks, a new routing algorithm based on ant colony optimization algorithm is proposed in [3]. In [4], an innovative parameter-adaptive strategy for ant colony optimization (ACO) algorithms based on controlling the convergence trajectory in decision space to follow any pre-specified path, aimed at finding the best possible solution within a given, and limited, computational budget is proposed. A novel nature-inspired meta-heuristic optimization algorithm, called Whale Optimization Algorithm (WOA), which mimics the social behavior of humpback whales is proposed in [5]. A list of heuristic or meta-heuristic algorithms that are developed by various researchers in past few years are provided in Table 1.

The main objective of cluster analysis is to find out groups of objects, called cluster(s) in a data set in such a way that the objects in a cluster are more similar than the objects in different clusters [6]. It is widely used for spatial data analysis, information retrieval, economic science (especially market research) and web data analysis etc. Some of the well-known clustering techniques are hierarchical, density based and partitional clustering [6]. In case of hierarchical clustering, the time complexity is quadratic in nature [7]. Once an object is assigned to a cluster, the hierarchical techniques do not reallocate the object, which may not be properly clustered in the previous stages of the assignments [8]. Whereas, density based clustering methods group neighboring objects into clusters based on local density conditions rather than proximity between objects. But this technique is unsuitable for high-dimensional data sets because of the curse of dimensionality phenomenon. Among the three clustering techniques, partitional clustering algorithms are widely used in real-time applications because of its simplicity and linear time complexity [7]. The partitional approach for data clustering is done by partitioning a collection of objects into a set of groups such that intra-cluster dissimilarity is minimum or inter-cluster dissimilarity is maximum. For simple analysis, any of the objectives can be optimized by an optimization algorithm. This inspires many researchers to handle the clustering problem as an optimization problem.

Many research articles are published by handling the clustering problem using different classical optimization algorithms. The classical algorithms such as Quadratic Programming (QP) [9], Dynamic Programming (DP) [10] and Lagrangian approach [11] fail to search the global optimal solutions due to high dimensionality of data sets and their local optimal searching techniques. The drawbacks of the numerical methods have forced the researchers to develop some alternatives. In the search of the alternatives, many

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2018.2812927, IEEE Transactions on Emerging Topics in Computing

nature inspired heuristic and meta-heuristic optimization algorithms are developed to obtain the optimal value. These are based on intelligence of insects and animals known as swarm intelligence algorithm. Many researchers are attracted towards different families of the algorithms such as Swarm Intelligence (SI) algorithms [12], [13], [14], [15], [16], [17], [18], [19] the evolutionary algorithms [20], [21], [22]. Many nature inspired algorithms are proposed, such as Social Spider Algorithm (SSA) to obtain a global optimal solution of different optimization problem [23]. In [24], Cuckoo Optimization Algorithm (COA) and fuzzy system are proposed for data clustering. In [25], a new hybrid Cat swarm optimisation (CSO) algorithm is proposed by incorporating Monte Carlo-based search equation and population centroid (PopC) operator with Gaussian probability distribution. In [25], Monte Carlo-based search equation is applied to enhance the diversity of the classical CSO algorithm. The population centroid (PopC) operator is used to prevent the classical CSO algorithm from trapping in local optima. In [26], a population-based algorithm inspired by the kidney process in the human body is proposed. A Grasshopper Optimisation Algorithm (GOA) is proposed in [27]. The GOA algorithm mathematically models and mimics the behaviour of grasshopper swarms in nature to solve various optimization problems. In [28], a Selective Refining Harmony Search inspired by the improvisation process of musicians is proposed in which a new harmony memory update has been utilized. A bio-inspired algorithm called Artificial Butterfly Optimization (ABO) algorithm is introduced in [29]. The ABO algorithm is based on the mate-finding strategy of some butterfly species. An Electro-Search (ES) algorithm inspired from the movement of electrons through the orbits around the nucleus of an atom is proposed in [30].

TABLE 1: Different Existing Classical Heuristic or Meta-Heuristic Optimization Algorithms

Algorithm	Inspired from	Proposed Year
PSO	Bird ock	1995
Marriage in Honey Bees Optimization Algorithm (MBO) [31]	Honey Bees	2001
Harmony Search(HS) [32]	Music player	2001
Articial Fish-Swarm Algorithm (AFSA) [33]	Fish swarm	2003
Termite Algorithm [34]	Termite colony	2005
ACO [35]	Ant colony	2006
ABC [36]	Honey Bee	2006
Wasp Swarm Algorithm [37]	Parasitic wasp	2007
Monkey Search [38]	Monkey	2007
Wolf pack search algorithm [39]	Wolf herd	2007
Biogeography Based Optimization [40]	mathematics of biogeography	2008
Bee Collecting Pollen Algorithm (BCPA) [41]	Bees	2008
Cuckoo Search (CS) [42]	Cuckoo	2009
Dolphin Partner Optimization (DPO) [43]	Dolphin	2009
Group Search Optimizer (GSO) [44]	Animal searching	2009
Bat-inspired Algorithm (BA) [45]	Bat herd	2010
Firey Algorithm (FA) [46]	Firey	2010
Hunting Search (HS) [47]	Group of animals	2010
Krill Herd (KH) [48]	Krill herd	2012
Fruit y Optimization Algorithm (FOA) [49]	Fruit y	2012
Dolphin Echolocation (DE) [50]	Dolphin	2013
Mine Blast Algorithm (MBA) [51]	Mine bomb explosion	2013
Social Based Algorithm (SBA) [52]	Human	2013
Group Counseling Optimization (GCO) [53]	Human	2014
Elephant Search Optimization [54]	Characteristics of elephant herds	2015
Whale Optimization Algorithm [5]	Hunting behavior of Whale	2016
Lion Optimization Algorithm (LOA) [55]	Lion's life style	2016

2 Novelty of the Paper

In this paper, an inspirational source based on human intelligence is considered for developing a new meta-heuristic optimization algorithm named as Class Topper Optimization (CTO) algorithm. Here, the information source is the learning behavior of students in a section to become the topper of the class. The students try to improve their performance at every stage of examinations. The knowledge improvement journey of a student to become the best is the flow of the proposed algorithm. The proposed algorithm is a population based meta-heuristic algorithm. To the knowledge of the present authors, there is no previous study on this subject in the literature of optimization algorithms. The student performance improves by gaining knowledge from the best student of their class. With this improvement factor in the student knowledge, one of the students become the topper of the class (global best). The objective is to improve the performance of the class topper in subsequent examinations. Every student tries to gain knowledge, which results improvements in the performance of the class topper. Two types of knowledge improvements are introduced in the proposed CTO algorithm such as: knowledge improvements in section level and in student level. The changes in learning happens because of these knowledge improvement concepts. The proposed CTO algorithm is validated by a complex optimization problem, i.e. clustering. The notations that are used throughout this paper is presented in Table 2.

TABLE 2: Notations

Symbol	Meaning
D	Data set with M dimensions
N	Total number of objects or data points
$R^{N\times M}$	Euclidean space with dimension $N \times M$
E_{max}	Maximum number of examinations or iterations
SE_{max}	Maximum number of sections
S_{max}	Total number of students
Π	Clustering
C_i	i^{th} cluster or i^{th} course
$D(X_1, Y_1)$	Euclidean distance between two M dimensional data points X_1 and Y_1
$SICD$	Sum of the intra-cluster distances
$TSICD$	Total $SICD$
$SICD(D, C)$	Objective function
ST	Section topper
CT	Class topper
$Scentroids^{(S,E)}$	Centroid values of the clusters that are obtained by student S in E^{th} examination
$STcentroids^{(SE,E)}$	Centroid values of the clusters that are obtained by section topper ST at SE^{th} section and E^{th} examination
$CTcentroids^{(E)}$	Centroid values of the clusters that are obtained by class topper CT in E^{th} examination
$I_1{}^{(SE,E)}$	Improvement in knowledge of ST of section SE in E^{th} examination
$I^{(SE,E)}$	Improvement in knowledge of student S of section SE in E^{th} examination
$ST_{pi}{}^{(SE,E)}$	Performance index of section topper of SE^{th} section in E^{th} examination
$CT_{pi}{}^{(E)}$	Performance index of class topper in E^{th} examination
$S_{pi}{}^{(S,E)}$	Performance index of S^{th} student in E^{th} examination
$I_{WF}{}^{E}$	Inertia weight factor at E^{th} examination
$I_{WF_{max}}$	Maximum inertia weight factor (value is equal to 0.5)
$I_{WF_{min}}$	Minimum inertia weight factor (value is equal to 0)
c	Acceleration coefficient in student level (value is equal to 2)
n_1	Uniform random number between 0 and 1 in section level
n_2	Uniform random number between 0 and 1 in student level
APE	Average percentage of error in classification

3 Proposed Class Topper Optimization (CTO) Algorithm

This optimization algorithm is inspired from the learning behavior of students in a specific class of a school. The students of same class compete with each other to be the best student or class topper of that class. This is a population based meta-heuristic optimization algorithm. Here, the global optima is achieved by learning in every stage. In the proposed algorithm, improvement in learning by the students, always happen towards the global optimal point (topper). Thus, gradually the algorithm converges to the global optimal solution. The algorithm is elaborated as follows. It is considered that the topper in a specific class of a school is the best student in study for that particular

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2018.2812927, IEEE Transactions on Emerging Topics in Computing

class. The topper is considered as the most learned person in that class. The CTO algorithm proposed in this paper is based on intelligence to obtain the class topper of a class. The algorithm works in three different stages. The three different stages are at the level of class, section and student. A graphical representation of the stages is shown in Fig. 1. Let us discuss in brief about learning at different stages.

Class level: In a school, there are different classes. A class

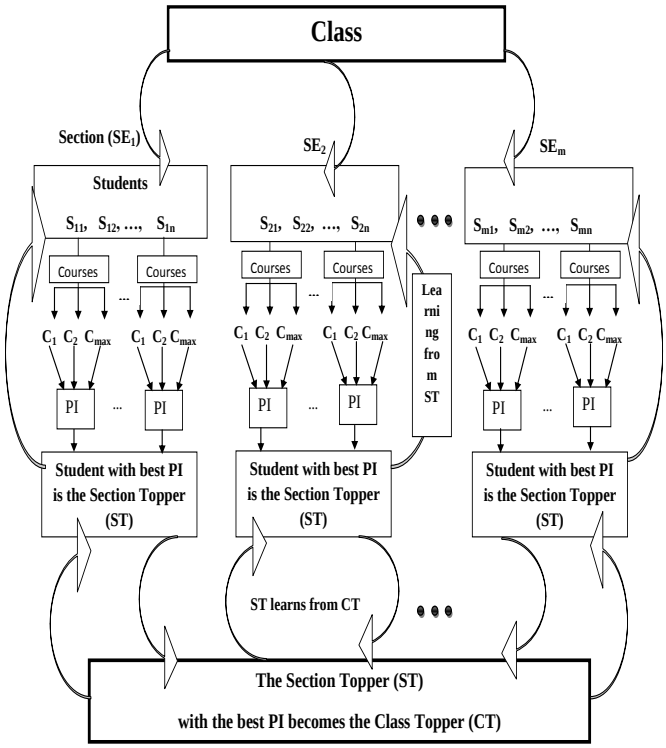

Fig. 1: Graphical view of the class topper learning concept

is divided into number of sections. The concept of making sections in a class is to make small groups. The students in these groups, i.e, sections can have better interaction with section topper (ST: best student in the section). This gives a steady improvement in the knowledge of the students (gives diverse and better partial solution). The best student of a class or the class topper (CT) of a class is the most learned person of all subjects in the class. The topper of the class definitely belongs to one section of that class. The transfer of knowledge from class topper to the students is limited to the section of which the class topper belongs to. But, as the STs are the best learned persons of the sections, they have the competition with CT and they will try to learn from CT. The learning will bring some improvements in the students performance.

Section (SE) level: A section is sub group of a class. The students from any section belong to the same class. The best student of a section is the section topper (ST). He may or may not be the class topper of a particular class. There is one limitation for the students other than the section topper to learn only from their respective ST. After every examination and evaluation process, the performance of the students in all sections may improve. This may leads to change in ST. The improvement occurs because of learning. As the ST is the best student of the section, he/she tries to learn from the CT.

Student (S) level: A particular class of a school is divided into several sections which consists of a set of students (search agents) with different level of knowledge (Performance Index (PI) or fitness value). All the students in the class are assigned same set of courses or subjects (dimension of the search space). Out of all the students from a class, the best student (with optimal fitness value) is the topper of the class. The score in each courses decides the performance (fitness value) of a student. Ideally, a student learns from the best student or section topper (ST) of its section only. The learning from the best student of the section has an impact on the other students of the section. This happens to bring an improvement in each student after every examination (evaluation of PI). In our evaluation process, each student will be given chance to improve after appearing in an examination. If the student will fail to improve in a particular examination, then his previous best result will be retained.

Examination (E): Examination is the process of evaluating PI of all the students. If performance of the students are not satisfactory, then the students may improve the PI by writing examinations. After the examination and evaluation process, if the performance of a particular student is found to be degraded than that of previous performance indices, then the previous best PI of the student will be considered for the future evaluation. From second examination onwards, ST learns from CT and the students of every section learn from their respective STs. This examination pattern (search process), will definitely lead to the best solution. Thus, PI of the class topper may be improved in successive examinations. The maximum number of examinations to improve the performance of students should be fixed.

Learning: Learning is a process to acquire knowledge to improve the performance of any individual. This may not happen for all the students in a section. But some of the students will learn definitely from the best student of the section. The students will not get a scope to learn from the class topper. Mathematically, the learning of a student is presented as follows:

Every student in each section learns from their respective ST as follows:

$$I^{(S,E+1)} = I_{WF} \times I^{(S,E)} + c \times n_2 \times \left({ST_{pi}}^{(SE,E)} - {S_{pi}}^{(S,E)} \right) \quad (1)$$

$${S_{pi}}^{(S,E+1)} = {S_{pi}}^{(S,E)} + I^{(S,E+1)} \quad (2)$$

As the section topper is the best student of the section, he/she must be enthusiastic to compete with the CT which gives learning to ST to improve.

ST learns from the CT as follows:

$${I_1}^{(SE,E+1)} = I_{WF} \times {I_1}^{(SE,E)} + c \times n_1 \times \left({CT_{pi}}^{(E)} - {ST_{pi}}^{(SE,E)} \right) \quad (3)$$

$${ST_{pi}}^{(SE,E+1)} = {ST_{pi}}^{(SE,E)} + {I_1}^{(SE,E+1)} \quad (4)$$

Here, learning means some knowledge improvements are added with the difference between PI of a student and PI of the best student (ST or CT). I and I_1 in the above equations are the knowledge improvements, that are introduced to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2018.2812927, IEEE Transactions on Emerging Topics in Computing

bring changes in students performance. In the improvement updating process some constants like, I_{WF}, c, n_1 and n_2 should be determined in advance. c represents the stochastic acceleration term that pull each student agent towards the optimal PI. It is required to select the inertia weight factor I_{WF}, such that the balance between global exploration and local exploration is maintained. In-order to enhance the convergence characteristic, I_{WF} is defined to decrease linearly:

$$I_{WF}{}^{E} = I_{WF\max} - \left(\frac{I_{WF\max} - I_{WF\min}}{E_{\max}} \times E \right) \quad (5)$$

In the proposed algorithm, the performance of the students is evaluated through examinations (iteration). After every examination, the ST and CT are selected and improvement factor for each student is calculated. This improvement factor brings better quality solution. This gives a global optimal solution. The proposed algorithm can be used for various real life complex optimization problem. To validate the proposed algorithm, we have considered a very complex clustering problem. To check diversity of the algorithm, we have also considered high dimensional data sets. In-order to have a clear vision and for easy understanding, a graphical view of the class topper concept is presented in Fig. 1. From Fig. 1, it is seen that a class is having set of sections and sections are having set of students. All the sections may or may not have equal number of students. The flowchart of the proposed CTO algorithm is shown in Fig. 2 and the proposed general CTO algorithm is presented as Algorithm 1.

Algorithm 1: Class Topper Optimization (CTO) Algorithm

1) **Input:** Initialize the number of section (SE_{max}), number of students in each section (S_{max}), number of courses allotted to each student (C) and number of examinations (E_{max}). Define other required parameters.
2) **Output:** Class Topper
3) ***BEGIN ALGORITHM***
4) /∗***Start examination***∗/
5) $For\ E = 1, 2, ..., E_{\max}$
6) /∗***For every section of a class***∗/
7) $For\ SE = 1, 2, ..., SE_{\max}$
8) /∗***Learning in section level***∗/
9) $If\ (E > 1)$
10) Section Topper (ST) learns from Class Topper (CT) using Eqn. 3 and 4
11) $End\ If$
12) $For\ S = 1, 2, ..., S_{\max}$
13) /∗***Learning in student level***∗/
14) $If\ (E > 1)$
15) Student S learns from ST using Eqn. 1 and 2
16) $End\ If$
17) /∗***Performance evaluation***∗/
18) Evaluate marks for all the courses
19) Calculate Performance Index (PI)
20) If the previous PI is better than the present PI, then keep the previous PI.
21) $End\ For$
22) /∗***Find Section Topper (ST)***∗/
23) Set the student with the best PI as the ST
24) $End\ For$

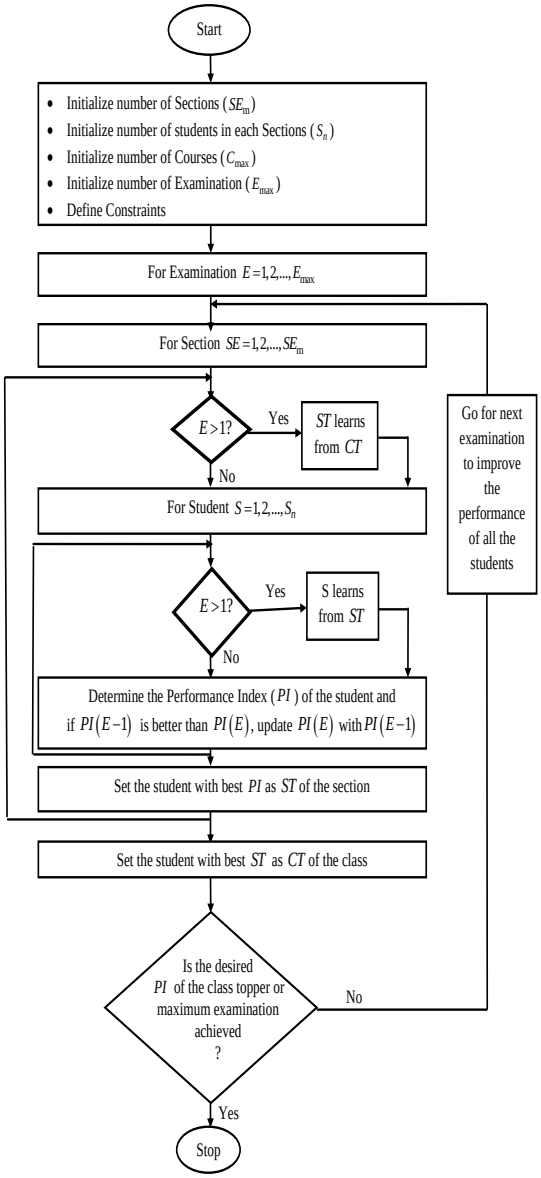

Fig. 2: Flow Chart of CTO Algorithm

25) /∗***Find Class Topper (CT)***∗/
26) Set the best ST as the CT
27) $End\ For$
28) Take the best CT out of all the examinations as the best student of the class
29) ***END ALGORITHM***

4 Data Clustering as Optimization Problem

The task of clustering is to make adhesive formation of data points or patterns or objects. Let D is the data set with N number of objects or patterns and each object is of M dimensional. D can be represented as $D = \{M_1, M_2, \ldots, M_N\}$, where $D \in R^{N \times M}$. Then, a clustering can be defined as $\Pi = \{C_1, C_2, \ldots, C_k\}$, such that $\bigcup_{i=1}^{k} C_i = D$, where C_i

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2018.2812927, IEEE Transactions on Emerging Topics in Computing

refers to the i^{th} cluster. The main objective of clustering is to find groups of data points in such a way that sum of the distances of the data points from their cluster centers i.e, SICD should be minimum. Any optimization algorithm can be used to minimize the SICD values. In this paper, the proposed optimization algorithm is used to solve single objective optimization problem for data clustering. Here, $SICD$ of all the existing clusters in a given data set is considered as the objective function. The mathematical expression for the objective function is given as follows:

$$SICD\left(D,C\right)=\sum_{i=1}^{n}\min_{j=1,2,\ldots k}D\left(o_i,c_j\right), \tag{6}$$

where $D\left(o_i,c_j\right)$ denotes the Euclidean distance between object o_i and center c_j, which can be represented as:

$$D\left(X_1,Y_1\right)=\sqrt{\sum_{i=1}^{M}\left(X_{1i}-Y_{1i}\right)^2}, \tag{7}$$

where X_1 and Y_1 are the vectors or objects with M dimension, X_{1i} and Y_{1i} are the i^{th} feature of X_1 and Y_1 respectively.

5 Mapping of CTO Algorithm for Data Clustering Problem

CTO algorithm can be used as an optimization tool to search the best solution in search space. To check the effectiveness of the proposed algorithm, a well-known clustering problem is considered. This algorithm is used for optimal assignment of data points to different clusters such that $SICD$ is minimum.

Mapping of CTO algorithm with clustering problem: The proposed CTO algorithm can be used for clustering problem. The algorithm is a meta-heuristic search algorithm. Searching for the class topper (best solution or best cluster centers) is the objective of the algorithm. The students of different sections of a class are the agents for searching the optimal solution (assigning data points to clusters such that $SICD$ is minimum). The number of courses taken by each student of a particular class is the number of clusters or number of classes present in the given data set. After each examination (iteration), the students will learn to improve the performance in the next examination. In subsequent examinations, the performance of the students ($SICD$ values of the clusters) will be monitored. In each iteration, the students will learn from the section topper or the class topper. Here in clustering, the learning means centroid values of the clusters will be updated in iterations. The purpose is to find optimal cluster center (centroid), which can represent the cluster in such a way that $SICD$ is minimum. In each iteration, the centers will be updated towards the optimal centroid. Therefore, the solution will converge to a global optimal solution. The class topper algorithm is mapped with data clustering problem. A graphical view of data clustering using CTO algorithm is presented in Fig. 3 and the mapped algorithm is presented as Algorithm 2 and is described in the next section.

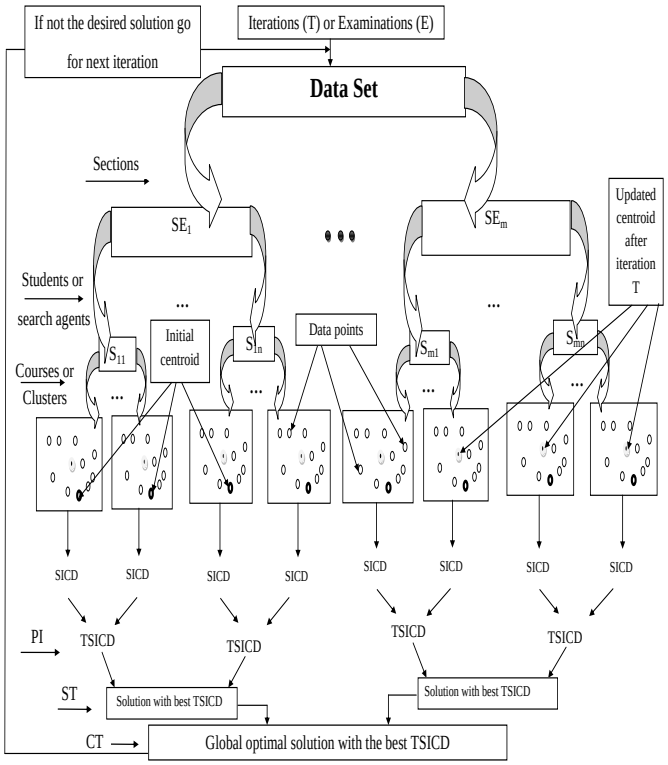

Fig. 3: Graphical representation of CTO algorithm for data clustering

Algorithm 2: CTO for Data Clustering

1) **Input:** Data set D with dimension $(N \times M)$, initialize number of sections (SE_{max}), number of students (agents) in each section (S_{max}), number of courses (C) or number of clusters and number of examinations ($E_{max} = T$) or number of iterations (T). Define other required constraints.
2) **Output:** An optimal clustering solution
3) ***BEGIN ALGORITHM***
4) $/*$***Start examination or iteration***$*/$
5) $For\ E = 1, 2, ..., E_{\max}$
6) $/*$***For every section of a class***$*/$
7) $For\ SE = 1, 2, ..., SE_{\max}$
8) $/*$***Learning in section level***$*/$
9) $If\ (E > 1)$
10) Section Topper (ST) learns from Class Topper (CT) using Eqn. 9 and 10
11) $End\ If$
12) $For\ S = 1, 2, ..., S_{\max}$
13) $/*$***Learning in student level***$*/$
14) $If\ (E > 1)$
15) Student S learns from ST using Eqn. 11 and 12
16) $End\ If$
17) $/*$***Initialization of centroid values for all the clusters or for all the courses***$*/$
18) $If\ (E == 1)$
19) Initialize centroids randomly from the given data set
20) $End\ If$
21) $/*$***Data allocation and*** $SICD$ ***or Performance Index (***PI ***calculation)***$*/$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2018.2812927, IEEE Transactions on Emerging Topics in Computing

22) Calculate distance matrix
23) Allocate data points to clusters based on distance
24) Find the mean of the clusters as new centroids
25) Calculate $SICD$
26) $If\,(SICD\,(E-1) < SICD\,(E))$
27) $\quad SICD\,(E) = SICD\,(E-1)$
28) $\quad Centroids\,(E) = Centroids\,(E-1)$
29) $End\ If$
30) $End\ For$
31) $/*$***Find Section Topper (ST)*** $*/$
32) Set the student with best $SICD$ as the ST and corresponding centroid values as $STcentroids$
33) $End\ For$
34) $/*$***Find Class Topper (CT)*** $*/$
35) Set best ST as CT and corresponding centroids as $CTcentroids$
36) $End\ For$
37) Get the best CT as optimal clustering solution
38) ***END ALGORITHM***

TABLE 3: Calculations of number of operations performed in Algorithm 2

Line number	Number of operations
5	$(E_{\max}+1)$
7	$(E_{\max} \times (SE_{\max}+1))$
9	$(E_{\max} \times SE_{\max})$
10	$(E_{\max} \times SE_{\max})$
12	$(E_{\max} \times SE_{\max} \times (S_{\max}+1))$
14	$(E_{\max} \times SE_{\max} \times S_{\max})$
15	$(E_{\max} \times SE_{\max} \times S_{\max})$
18	$(E_{\max} \times SE_{\max} \times S_{\max})$
19	$(E_{\max} \times SE_{\max} \times S_{\max})$
22	$(E_{\max} \times SE_{\max} \times S_{\max}) \times (N \times M)$
23	$(E_{\max} \times SE_{\max} \times S_{\max}) \times (N \times M)$
24	$(E_{\max} \times SE_{\max} \times S_{\max}) \times (N \times M)$
25	$(E_{\max} \times SE_{\max} \times S_{\max}) \times (N \times M)$
26	$(E_{\max} \times SE_{\max} \times S_{\max}) \times (N \times M)$
27	$(E_{\max} \times SE_{\max} \times S_{\max}) \times (N \times M)$
28	$(E_{\max} \times SE_{\max} \times S_{\max}) \times (N \times M)$
32	$(E_{\max} \times SE_{\max})$
35	E_{max}

6 Description of Algorithm 2

6.1 Initialization

All the required parameters for first examination for the proposed CTO algorithm are initialized here. The centroid values of the clusters for every student agent of all the sections for the first examination or first iteration are randomly initialized from the given data set. The data points are allocated to the initialized cluster centers. From second examination (iteration) onwards, the centroids are obtained from the generated solution of the previous iteration.

6.2 Assigning data points to clusters and generating partial solutions

In this stage, every student agent studies and gives examinations for all the courses that are assigned to them. In other words, a student allocates all the data points to clusters centers and generates a partial solution. At first distance matrix is calculated. Then data points are assigned based on their distance from the cluster centers. In every iteration data will be assigned to the closest cluster center. Thus, every student generates a partial solution by creating a set of clusters of the given data set.

6.3 Evaluating quality (PI or TSICD) of the partial solutions

After the data allocation, qualities of the discovered solutions are determined. At first, new centroid values of the obtained clusters are calculated by taking the mean of the clusters. Then, $SICD$ and total $SICD$ or $TSICD(PI)$ values are calculated. The $TSICD$ is defined as follows:

$$TSICD(S) = \sum_{C=1}^{C_{max}} SICD\,(C,S). \tag{8}$$

If $PI(E-1)$ is better than $PI(E)$, update $PI(E)$ with $PI(E-1)$. In other sense, if the clustering solution obtained in the previous iteration is better than the present iteration, then keep the previous solution.

6.4 Find section topper and class topper

After determining PI ($TSICD$) values for all the clusters of all the agents or for all the students of all the sections, PI of section toper (ST) and class topper (CT) will be determined. The student having best clustering solution or who scores best in a section will be set as a ST. After getting ST for all the sections, the ST with best clustering solution will be set as CT of the class.

Now, if the desired optimal solution or desired PI is obtained or maximum examination (iteration) reached, then stop the process. Otherwise, go for next iteration to search for better solution with the help of learning. Learning happens in two levels which are described in the following section.

6.5 Learning in section level and student level

Here, in the present problem, student agents search for optimal solution, i.e, cluster centers or centroids of the clustering solution or clusters. At the starting, cluster centers for all the student agents are initialized randomly from the data set and after that centers get updated in every iteration. Hence, the learning task in the section level and student level for the data clustering problems are performed as follows:

$$\begin{aligned} {I_1}^{(SE,E+1)} &= I_{WF} \times {I_1}^{(SE,E)} + c \times n_1 \\ &\times \left(CTcentroids^{(E)} - STcentroids^{(SE,E)}\right). \end{aligned} \tag{9}$$

$$STcentroids^{(SE,E+1)} = STcentroids^{(SE,E)} + {I_1}^{(SE,E+1)}. \tag{10}$$

$$\begin{aligned} I^{(S,E+1)} &= I_{WF} \times I^{(S,E)} + c \times n_2 \\ &\times \left(STcentroids^{(SE,E)} - Scentroids^{(S,E)}\right). \end{aligned} \tag{11}$$

$$Scentroids^{(S,E+1)} = Scentroids^{(S,E)} + I^{(S,E+1)}. \tag{12}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2018.2812927, IEEE Transactions on Emerging Topics in Computing

7 Result Analysis

In this paper, the proposed CTO algorithm is applied to find clustering solution for well known data sets. Five different standard data sets are used to analyze the performance of the algorithm. Here, in Algorithm 2, the parameter I_{WF} is the inertia weight factor which is actually the weightage that are given on the previous improvement. If the value of the weightage is more than 0.5, the convergence will definitely be faster, but the algorithm may stuck at local optima. On the other hand, if the weightage is less than 0.5, the convergence rate will slow down. To handle this issue, I_{WF} is assumed be in the range of 0 to 0.5. The value of the I_{WF} is fine tuned at 0.5 to obtain the global optimal with faster convergence. The parameter c is a constant term, which is actually a stochastic acceleration coefficient, which drag the searching agents towards the best solution. If c is greater than 2, then it may be seen that the improvement quickly explodes to large values, especially for students having poor performance compared to their section topper or class topper. Which may cause suddenly a large changes in the students performance index. In the other hand, if c value is very less, e.g, $c < 1.5$, the changes in the performance index may be very less. In-order to tackle this scenario, the c value should be fine tuned. Here, in this algorithm, better quality solutions are obtained by considering $c = 2.0$ (for the undertaken clustering problem, the position of the new centroids may move out of the search space or may be very close to the previous centroids depending upon the values of c and I_{WF}). The random variables n_1 and n_2 are taken in the range of (0,1) which bring some randomness in the performance improvements.

$SICD$ value is used as the performance measure of the algorithm with different existing algorithms. In Table 5 and 6, a performance analysis is carried out by varying the number of students and number of sections for all the five data sets. A graphical analysis for the same is presented in Fig. 4 and 5 for Iris and Wine data set. From the Fig. 4 and 5, it is observed that if the number of students and number of sections are increasing, then quality of the solution improves. But, after increasing the number students and sections to a certain value, e.g, SE=5 and S=60 for Iris and SE=10, S=50 for Wine data, the improvement in the quality of the solution is not significant. Because, after reaching a particular number of searching agents which can cover the entire search space, the searching solution gets saturated and including a more number of agents may provide the same solution. One may observe the same behavior of the algorithm from the analysis of other data sets presented in Table 5 and 6.

Here, it is required to find an optimal number of student agents and number of sections for searching the solution space. For a simple analysis, in each section equal number of students are considered here. It is obvious that, in a class, if the number of sections and number of students in each section are more, then after successive examinations the class topper may provide better performance. From the analysis, it is seen that, based on the dimension of the data sets, the number of search agents should be tuned. Therefore, a high dimensional data set, e.g, HV data set, needs more number of searching agents (S=60 and SE=15 for HV data in Table 5) to find an optimal solution compared to a data set with lower dimension.

To compare the efficiency of the proposed algorithm with other well known existing algorithms, an analysis is done and presented in Table 7. Here, in Table 7, the parameter settings for the proposed algorithm are chosen from the parameters in Table 5 and 6 for which best quality results were obtained. One can observe that the proposed algorithm provides superior quality solution compared to some well known existing heuristic and meta-heuristic optimization algorithms. To study the run-time behavior and convergence rate of the proposed algorithm, an asymptotic analysis is presented in the following section.

TABLE 4: Description of the data sets [56]

Data set	Number of clusters or centers	Number of features or attributes	Number of data objects or patterns
Iris	3	4	150(50,50,50)
Wine	3	13	178(59,71,48)
WBreast-Cancer	2	9	699(458,241)
CMC	3	9	1473(629,334,510)
Hill-Valley(HV)	2	101	606(305,301)

TABLE 5: Analysing $SICD$ by changing number of students (S)

Data set	Other parameters	S value	SICD
Iris	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$, $SE = 5$ and $E_{max} = T = 100$	20	96.10
		30	95.11
		40	95.01
		50	94.91
		60	94.50
Wine	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$, $SE = 10$ and $E_{max} = T = 100$	20	16317.01
		30	16310.00
		40	16298.79
		50	16262.01
		60	16262.01
Cancer	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$, $SE = 10$ and $E_{max} = T = 100$	20	3226.39
		30	3001.11
		40	2960.10
		50	2960.10
		60	2960.10
CMC	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$, $SE = 10$ and $E_{max} = T = 100$	20	5770.10
		30	5690.51
		40	5688.31
		50	5678.04
		60	5678.04
HV	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$, $SE = 15$ and $E_{max} = T = 100$	20	63930100.48
		30	63929700.31
		40	63929455.04
		50	63929289.30
		60	63929280.60

TABLE 6: Analysing $SICD$ by changing number of sections (SE)

Data set	Other parameters	SE value	S value	SICD
Iris	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$ and $E_{max} = T = 100$	5	60	94.50
		10	20	95.06
		15	40	94.50
		20	30	94.50
		25	50	94.50
Wine	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$ and $E_{max} = T = 100$	5	60	1700.00
		10	50	16262.00
		15	40	16262.00
		20	50	16262.00
		25	30	16262.00
Cancer	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$ and $E_{max} = T = 100$	5	60	3310.00
		10	40	2960.10
		15	50	2960.10
		20	60	2960.10
		25	50	2960.10
CMC	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$ and $E_{max} = T = 100$	5	60	5702.00
		10	50	5678.04
		15	50	5678.04
		20	60	5678.04
		25	40	5678.04
HV	$c = 2$, $I_{WF\max} = 0.5$, $I_{WF\min} = 0$ and $E_{max} = T = 100$	5	60	72789350.09
		10	60	65909989.60
		15	60	63929280.60
		20	50	63929280.60
		25	50	63929280.60

8 Asymptotic analysis of the proposed algorithm

Asymptotic analysis of an algorithm refers to define the mathematical framing of its run-time performance [63]. In

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2018.2812927, IEEE Transactions on Emerging Topics in Computing

TABLE 7: Comparative analysis of the $SICD$ values of the proposed algorithm with some existing baseline and meta-heuristic algorithms

Data set	Measure	Classical-PSO [57]	K-means [58]	PSO [59]	K-NM-PSO [60]	K-PSO [61]	IBCOCLUST [62]	IBKCLUST [62]	KIBCLUST [62]	CTO
Iris	Best	96.78	97.33	96.66	96.66	96.66	97.22	97.33	96.40	94.50
	Average	103.62	106.05	103.51	96.67	96.76	97.27	97.33	96.40	96.21
	Std	4.48	14.63	4.45	4.21	4.33	4.31	4.30	4.30	4.40
Wine	Best	16294.21	16555.68	16294.00	16292.00	16292.00	16460.00	16460.55	16453.28	16262.00
	Average	16311.50	18161.00	16311.00	16293.00	16296.00	16460.55	16460.90	16460.60	16293.98
	Std	108.01	793.21	107.18	107.02	109.13	108.43	108.39	108.37	106.02
Cancer	Best	2976.41	2987.00	2976.30	2964.50	2964.50	2976.24	2976.06	2980.15	2960.10
	Average	3337.09	2988.30	3334.60	2964.70	2965.80	2976.89	2976.33	2980.15	2965.13
	Std	2.31	251.14	2.27	3.40	3.49	3.34	3.30	3.29	3.17
CMC	Best	5694.28	5842.20	5694.20	5689.00	5691.88	5691.60	5690.43	5690.30	5678.04
	Average	5728.06	5893.60	5729.30	5731.70	5725.60	5725.20	5727.01	5727.50	5723.05
	Std	40.00	47.16	40.24	39.02	38.78	38.87	38.79	38.80	38.76
HV	Best	63929350.40	63929500.11	63929490.03	63929312.40	63929327.30	63929328.22	63929326.42	63929326.29	63929280.60
	Average	66214090.02	66214102.10	66214099.00	66214110.03	66214100.11	66214104.13	66214102.60	66214102.58	66214050.00
	Std	358300.10	358450.00	358401.10	358389.21	358392.43	358391.65	358393.48	358393.39	358250.02

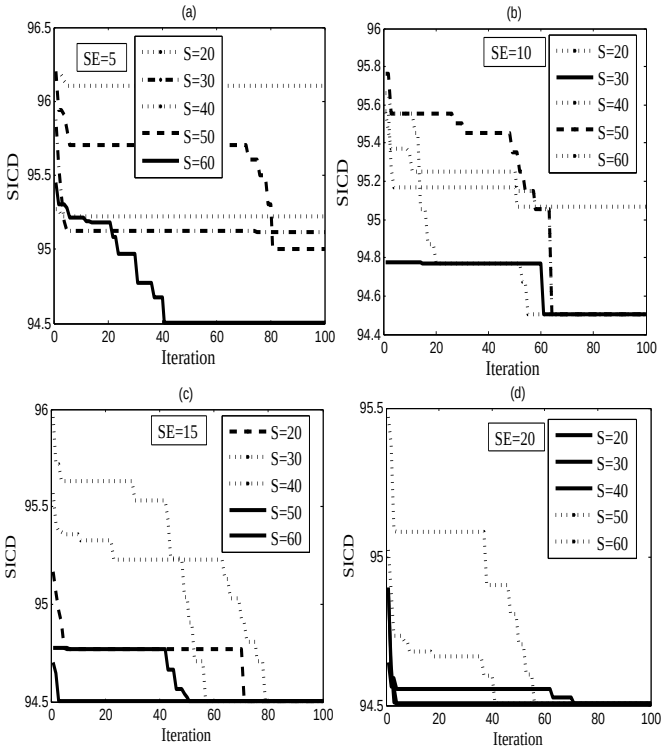

Fig. 4: $SICD$ analysis of the proposed CTO algorithm for Iris data by changing number of students (S = 20, 30, 40, 50 and 60) and number of sections (SE): (a) $SE = 5$, (b) $SE = 10$, (c) $SE = 15$ and (d) $SE = 20$.

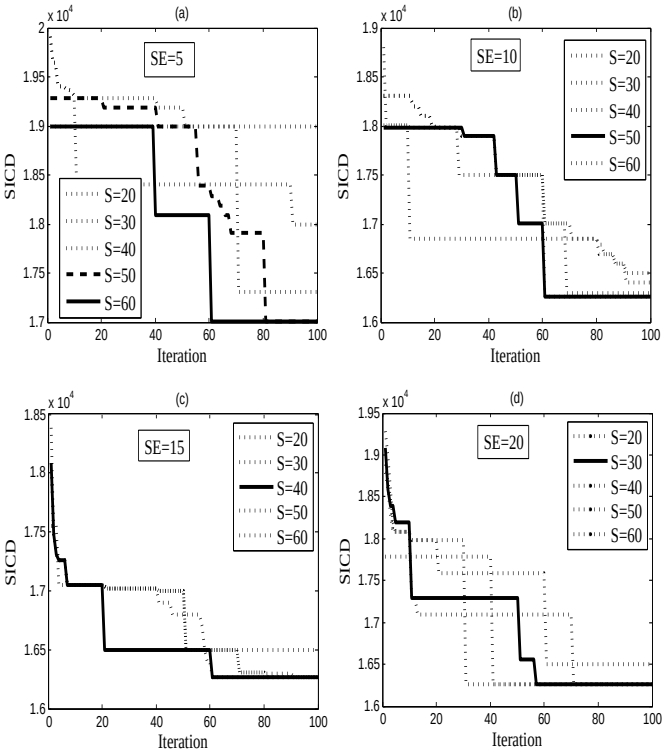

Fig. 5: $SICD$ analysis of the proposed CTO algorithm for Wine data by changing number of students (S = 20, 30, 40, 50 and 60) and number of sections (SE): (a) $SE = 5$, (b) $SE = 10$, (c) $SE = 15$ and (d) $SE = 20$.

asymptotic analysis, we evaluate the performance of an algorithm in terms of input size. We calculate, the influence of input size with the running time of the algorithm. The proposed CTO algorithm depends on some input parameters, such as number of examinations (E_{max}=E) which actually denotes the number of iterations (T), number of sections (SE_{max}=SE), number of students (S_{max}=S), number of data points (N) and number of attributes (M) of the data set. The total number of operations (TO) that are required to execute the proposed CTO algorithm for data clustering are determined from Table 3 of Algorithm 2 as follows:

$$\begin{aligned} TO\left(T,SE,S,N,M\right) &= (7\times T\times SE\times S\times N\times M) \\ &\quad +(5\times T\times SE\times S) \\ &\quad +(5\times T\times SE) \\ &\quad +(3\times T)+1. \end{aligned} \tag{13}$$

For the run time performance analysis of Algorithm 2 all parameters are assumed to be equal for worst case scenario. Then, (13) may be written as a function $f\left(n\right)$.

$$f\left(n\right)=7n^{5}+5n^{3}+5n^{2}+3n+1. \tag{14}$$

We assume that $f\left(n\right)$ is $big-O$ of n^{5},i.e. $g\left(n\right)=n^{5}$. Now, $f\left(n\right)=O\left(g\left(n\right)\right)$, if and only if, there are two positive constants c and n_{0} such that

$$\left|f\left(n\right)\right|\leq c\left|g\left(n\right)\right|, \tag{15}$$

for all $n\geq n_{0}$, here $f\left(n\right)$ is nonnegative, (15) can be written as:

$$0\leq f\left(n\right)\leq cg\left(n\right), \tag{16}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2018.2812927, IEEE Transactions on Emerging Topics in Computing

for all $n \geq n_0$. c can be chosen as 21 by adding all the coefficients of (14). Then if $n \geq n_0 = 1$,

$$7n^5 + 5n^3 + 5n^2 + 3n + 1 \leq 7n^5 + 5n^5 + 5n^5 + 3n^5 + n^5,$$

thus $7n^5 + 5n^3 + 5n^2 + 3n + 1 = O\left(n^5\right)$. Therefore, $f\left(n\right) = O\left(g\left(n\right)\right) = O\left(n^5\right)$. As n increases, $f\left(n\right)$ will always be less than or equal to $g\left(n\right)$. In other words, $g\left(n\right)$ is an asymptotic upper bound on $f\left(n\right)$. It indicates that the algorithm runs in polynomial time. The running time of the CTO algorithm (Algorithm 2) with respect to the input parameters is shown in Fig. 6. From Fig. 6, it is observed that $f\left(n\right)$ is always less than or equal to $g\left(n\right)$. Now, from the analysis, it is inferred that the proposed CTO algorithm can be applied for real world data clustering problems.

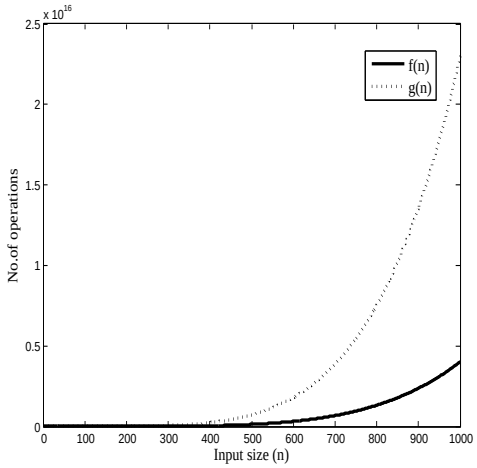

Fig. 6: Asymptotic behavior of the proposed algorithm

Comparative analysis of running time: Now, the performance of the proposed algorithm is compared with some existing algorithms with respect to time complexity and is presented in Table 8. From Table 5, 6, 7 and 8, it can be observed that the proposed CTO algorithm provides better quality solution compared to IBCOCLUST and PSO with the same running behaviors. Now, from the analysis, it is inferred that the proposed CTO can also be applied for real world data clustering problems. To verify the performance of the proposed algorithm further, a misclassification rate or classification error percentage analysis is presented in the following section.

TABLE 8: A Comparative Time Complexity Analysis

Algorithm	Time Complexity (in worst case)
Classical PSO [57]	$O\left(n^5\right)$
IBCOCLUST [62]	$O\left(n^5\right)$
CTO	$O\left(n^5\right)$

9 Performance analysis with respect to classification error percentage

The performance evaluation of a clustering or classification task can be done by various performance measure indices, e.g accuracy, sensitivity (recall), precision, F-measure and error rate [6] etc. The accuracy measure actually reflects how well the classifier recognizes tuples of the different classes. But, it is known that the accuracy is most effective when the class distribution is relatively balanced [6]. Now, for class imbalanced problems, where the main class of interest is rare. That means, the data set distribution reflects a significant majority of negative class and a minority of positive class. In that case, the classifier may not properly classify the positive tuples as positive and negative tuples as negative. Recall, precision and combination of both, i.e, F-measure can be used in this regard. Although all the measures are biased in some way [6], [64], here, for a simple analysis and to evaluate the classification error percentage, error rate or misclassification rate is considered.

The Percentage of Error (PE) is calculated to study the percentage of incorrectly classified data for test data sets. It is calculated from incorrectly assigned data with total test data. The PE is described as follows:

$$PE = \frac{s}{n} \times 100,$$

where s is the total number of incorrectly classified data tuples and n is the total size of test data set. The Average Percentage of Error (APE) analysis from the obtained PE values are calculated with multiple runs of the algorithms. The APE analysis is presented in Table 9 to cross validate the performance of the proposed algorithm compared to some existing algorithms. From Table 9, it is observed that the

TABLE 9: APE values after multiple runs

Data set	$PSO - \psi_1$ [65]	$PSO - \psi_2$ [65]	$PSO - \psi_3$ [65]	Classical-PSO [57]	IBCOCLUST [62]	K-means [58]	K-NM-PSO [60]	CTO
Iris	3.68	2.63	5.26	4.10	3.96	4.50	3.50	2.60
Wine	6.22	2.22	2.88	3.12	3.01	3.45	2.20	2.18
Cancer	3.49	5.80	3.49	3.49	3.99	4.01	3.30	3.12
CMC	3.93	3.90	3.87	3.90	3.88	4.12	3.81	3.79
HV	31.00	30.98	30.82	31.01	32.43	35.03	30.93	30.67

APE values of the proposed algorithm is better than the $PSO - \Psi_1$, $PSO - \Psi_2$, $PSO - \Psi_3$ and classical-PSO. From Table 7, it is observed that the proposed CTO algorithm provides better quality solutions compared the existing algorithms, which results a lower APE values in classification.

10 Conclusion, discussion and future scope

A new class toper optimization algorithm is proposed in this paper. The performance of the algorithm is analysed by a data clustering problem. From the analysis, this algorithm is found effective to obtain global optimal solution. A comparative analysis of the proposed CTO algorithm with some of the well known existing heuristic or meta-heuristic optimization algorithms are done using five well known standard data sets. It is observed from the previous SICD analysis, APE analysis and running time comparison that the proposed algorithm performs same or better compared to some existing algorithms and provides faster convergence.

It performs parallel search which is pretty good for data sets with higher dimension. The proposed CTO algorithm does not stuck in local optima. The algorithm provides better quality solutions compared to some well known existing methods with the same running behavior. But, the proposed algorithm fails to give better solution for non-spherical data. Because, the searching agents in the CTO, search the solution space by updating the positions. But

in non-spherical data, position based updating fails to give suitable results, for example, flag data set from UCI machine learning repository [56]. But, from the analysis, it is observed that the CTO algorithm provides global optimal solution with faster convergence.

Extensive experimental studies show that the proposed algorithm can efficiently deal with real-time data clustering problems. The algorithm can also be used for other optimization based real world problems.

REFERENCES

[1] H. Mo, L. Liu, and J. Zhao, "A new magnetotactic bacteria optimization algorithm based on moment migration," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 14, no. 1, pp. 15–26, 2017.

[2] S. H. A. Kaboli, J. Selvaraj, and N. Rahim, "Rain-fall optimization algorithm: A population based algorithm for solving constrained optimization problems," *Journal of Computational Science*, vol. 19, pp. 31–42, 2017.

[3] Y. Sun, W. Dong, and Y. Chen, "An improved routing algorithm based on ant colony optimization in wireless sensor networks," *IEEE Communications Letters*, 2017.

[4] F. Zheng, A. Zecchin, J. Newman, H. Maier, and G. Dandy, "An adaptive convergence-trajectory controlled ant colony optimization algorithm with application to water distribution system design problems," *IEEE Transactions on Evolutionary Computation*, 2017.

[5] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[6] M. K. Jiawei Han, "Data mining:concepts and techniques," *Elsevier*, 2006.

[7] M. Steinbach, G. Karypis, V. Kumar *et al.*, "A comparison of document clustering techniques," in *KDD workshop on text mining*, vol. 400, no. 1. Boston, 2000, pp. 525–526.

[8] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[9] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics (NRL)*, vol. 3, no. 1-2, pp. 95–110, 1956.

[10] R. Bellman, *Dynamic programming*. Courier Corporation, 2013.

[11] D. P. Bertsekas, *Nonlinear programming*. Athena scientific Belmont, 1999.

[12] Q. Ni, Q. Pan, H. Du, C. Cao, and Y. Zhai, "A novel cluster head selection algorithm based on fuzzy clustering and particle swarm optimization," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 14, no. 1, pp. 76–84, 2017.

[13] A. Chakraborty and A. K. Kar, "Swarm intelligence: A review of algorithms," in *Nature-Inspired Computing and Optimization*. Springer, 2017, pp. 475–494.

[14] A. P. Piotrowski, M. J. Napiorkowski, J. J. Napiorkowski, and P. M. Rowinski, "Swarm intelligence and evolutionary algorithms: Performance versus speed," *Information Sciences*, vol. 384, pp. 34–85, 2017.

[15] D. Van der Merwe and A. P. Engelbrecht, "Data clustering using particle swarm optimization," in *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, vol. 1. IEEE, 2003, pp. 215–220.

[16] F. Yang, T. Sun, and C. Zhang, "An efficient hybrid data clustering method based on k-harmonic means and particle swarm optimization," *Expert Systems with Applications*, vol. 36, no. 6, pp. 9847–9852, 2009.

[17] S. Rana, S. Jasola, and R. Kumar, "A review on particle swarm optimization algorithms and their applications to data clustering," *Artificial Intelligence Review*, vol. 35, no. 3, pp. 211–222, 2011.

[18] I. Fister Jr, X.-S. Yang, I. Fister, J. Brest, and D. Fister, "A brief review of nature-inspired algorithms for optimization," *arXiv preprint arXiv:1307.4186*, 2013.

[19] X. Min, L. Liu, Y. He, X. Gong, S. Fong, Q. Xu, and K. K. Wong, "Benchmarking swarm intelligence clustering algorithms with case study of medical data," *Computerized Medical Imaging and Graphics*, 2016.

[20] D. Simon, *Evolutionary optimization algorithms*. John Wiley & Sons, 2013.

[21] J. Branke, M. Asafuddoula, K. S. Bhattacharjee, and T. Ray, "Efficient use of partially converged simulations in evolutionary optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 1, pp. 52–64, 2017.

[22] M. Mavrovouniotis, C. Li, and S. Yang, "A survey of swarm intelligence for dynamic optimization: algorithms and applications," *Swarm and Evolutionary Computation*, vol. 33, pp. 1–17, 2017.

[23] J. James and V. O. Li, "A social spider algorithm for global optimization," *Applied Soft Computing*, vol. 30, pp. 614–627, 2015.

[24] E. Amiri and S. Mahmoudi, "Efficient protocol for data clustering by fuzzy cuckoo optimization algorithm," *Applied Soft Computing*, vol. 41, pp. 15–21, 2016.

[25] Y. Kumar and G. Sahoo, "Gaussian cat swarm optimisation algorithm based on monte carlo method for data clustering," *International Journal of Computational Science and Engineering*, vol. 14, no. 2, pp. 198–210, 2017.

[26] N. S. Jaddi, J. Alvankarian, and S. Abdullah, "Kidney-inspired algorithm for optimization problems," *Communications in Nonlinear Science and Numerical Simulation*, vol. 42, pp. 358–369, 2017.

[27] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: Theory and application," *Advances in Engineering Software*, vol. 105, pp. 30–47, 2017.

[28] M. Shabani, S. A. Mirroshandel, and H. Asheri, "Selective refining harmony search: A new optimization algorithm," *Expert Systems with Applications*, vol. 81, pp. 423–443, 2017.

[29] X. Qi, Y. Zhu, and H. Zhang, "A new meta-heuristic butterfly-inspired algorithm," *Journal of Computational Science*, 2017.

[30] A. Tabari and A. Ahmad, "A new optimization method: Electro-search algorithm," *Computers & Chemical Engineering*, vol. 103, pp. 1–11, 2017.

[31] H. A. Abbass, "Mbo: Marriage in honey bees optimization-a haplometrosis polygynous swarming approach," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 1. IEEE, 2001, pp. 207–214.

[32] Z. W. Geem, J. H. Kim, and G. Loganathan, "A new heuristic optimization algorithm: harmony search," *simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[33] X. Li, "A new intelligent optimization-artificial fish swarm algorithm," *Doctor thesis, Zhejiang University of Zhejiang, China*, 2003.

[34] R. Martin and W. Stephen, "Termite: A swarm intelligent routing algorithm for mobilewireless ad-hoc networks," *Stigmergic optimization*, pp. 155–184, 2006.

[35] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE computational intelligence magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[36] B. Basturk and D. Karaboga, "An artificial bee colony (abc) algorithm for numeric function optimization," in *IEEE swarm intelligence symposium*, vol. 8, no. 1, 2006, pp. 687–697.

[37] P. C. Pinto, T. A. Runkler, and J. M. Sousa, "Wasp swarm algorithm for dynamic max-sat problems," in *International Conference on Adaptive and Natural Computing Algorithms*. Springer, 2007, pp. 350–357.

[38] A. Mucherino and O. Seref, "Monkey search: a novel metaheuristic search for global optimization," in *AIP conference proceedings*, vol. 953, no. 1. AIP, 2007, pp. 162–173.

[39] C. Yang, X. Tu, and J. Chen, "Algorithm of marriage in honey bees optimization based on the wolf pack search," in *Intelligent Pervasive Computing, 2007. IPC. The 2007 International Conference on*. IEEE, 2007, pp. 462–467.

[40] D. Simon, "Biogeography-based optimization," *IEEE transactions on evolutionary computation*, vol. 12, no. 6, pp. 702–713, 2008.

[41] X. Lu and Y. Zhou, "A novel global convergence algorithm: bee collecting pollen algorithm," *Advanced intelligent computing theories and applications. With aspects of artificial intelligence*, pp. 518–525, 2008.

[42] X.-S. Yang and S. Deb, "Cuckoo search via lévy flights," in *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*. IEEE, 2009, pp. 210–214.

[43] Y. Shiqin, J. Jianjun, and Y. Guangxing, "A dolphin partner optimization," in *Intelligent Systems, 2009. GCIS'09. WRI Global Congress on*, vol. 1. IEEE, 2009, pp. 124–128.

[44] S. He, Q. H. Wu, and J. Saunders, "Group search optimizer: an optimization algorithm inspired by animal searching behavior," *IEEE transactions on evolutionary computation*, vol. 13, no. 5, pp. 973–990, 2009.

[45] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pp. 65–74, 2010.

[46] ——, "Firefly algorithm, stochastic test functions and design optimisation," *International Journal of Bio-Inspired Computation*, vol. 2, no. 2, pp. 78–84, 2010.

[47] R. Oftadeh, M. Mahjoob, and M. Shariatpanahi, "A novel meta-heuristic optimization algorithm inspired by group hunting of animals: Hunting search," *Computers & Mathematics with Applications*, vol. 60, no. 7, pp. 2087–2098, 2010.

[48] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831–4845, 2012.

[49] W.-T. Pan, "A new fruit fly optimization algorithm: taking the financial distress model as an example," *Knowledge-Based Systems*, vol. 26, pp. 69–74, 2012.

[50] A. Kaveh and N. Farhoudi, "A new optimization method: dolphin echolocation," *Advances in Engineering Software*, vol. 59, pp. 53–70, 2013.

[51] A. Sadollah, A. Bahreininejad, H. Eskandar, and M. Hamdi, "Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems," *Applied Soft Computing*, vol. 13, no. 5, pp. 2592–2612, 2013.

[52] F. Ramezani and S. Lotfi, "Social-based algorithm (sba)," *Applied Soft Computing*, vol. 13, no. 5, pp. 2837–2856, 2013.

[53] M. Eita and M. Fahmy, "Group counseling optimization," *Applied Soft Computing*, vol. 22, pp. 585–604, 2014.

[54] S. Deb, S. Fong, and Z. Tian, "Elephant search algorithm for optimization problems," in *Digital Information Management (ICDIM), 2015 Tenth International Conference on*. IEEE, 2015, pp. 249–255.

[55] M. Yazdani and F. Jolai, "Lion optimization algorithm (loa): a nature-inspired metaheuristic algorithm," *Journal of computational design and engineering*, vol. 3, no. 1, pp. 24–36, 2016.

[56] M. Lichman, "Uci machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[57] J. Kennedy and R. Eberhart, "Particle swarm optimization (pso)," in *Proc. International Conference on Neural Networks, Perth, Australia*. IEEE, 1995, pp. 1942–1948.

[58] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14, 1967, pp. 281–297.

[59] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of machine learning*. Springer, 2011, pp. 760–766.

[60] S.-K. S. Fan, Y.-c. Liang, and E. Zahara, "Hybrid simplex search and particle swarm optimization for the global optimization of multimodal functions," *Engineering Optimization*, vol. 36, no. 4, pp. 401–418, 2004.

[61] A. Ahmadyfard and H. Modares, "Combining pso and k-means to enhance data clustering," in *Telecommunications, 2008. IST 2008. International Symposium on*. IEEE, 2008, pp. 688–691.

[62] R. Forsati, A. Keikha, and M. Shamsfard, "An improved bee colony optimization algorithm with an application to document clustering," *Neurocomputing*, vol. 159, pp. 9–26, 2015.

[63] T. H. Cormen, *Introduction to algorithms*. MIT press, 2009.

[64] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[65] I. De Falco, A. Della Cioppa, and E. Tarantino, "Facing classification problems with particle swarm optimization," *Applied Soft Computing*, vol. 7, no. 3, pp. 652–658, 2007.

Pranesh Das is a PhD scholar in the Department of Computer Science and Engineering in National Institute of Technology Nagaland, India. He Received his M.Tech degree in Computer Science and Engineering from National Institute of Technology Rourkela, India in 2013. He did his B.Tech in Information Technology from Maulana Abul Kalam Azad University of Technology (West Bengal, India) in 2007. From 2008 to 2011, he worked as a Lecturer in the Department of Information Technology, BIET, Suri, West Bengal, India. Since 2013 to 2016 he worked as an Assistant professor in the Department of Computer Science and Engineering in National Institute of Technology Nagaland, India. His research interests include Data Mining, machine learning, soft computing and optimization algorithms.

Dushmanta Kumar Das is Member of IEEE. He received his B.Tech in Electronics and Instrumentation Engineering from BPUT, Odisha, India. He received M.Tech and Ph.D. degrees in Control System Engineering from the National Institute of Technology Rourkela, India in 2010 and 2015 respectively. Since 2013, he has been with National Institute of Technology Nagaland, India as an Assistant Professor. His research interests include soft computing, machine learning, Meta-heuristic optimization techniques and application of optimization techniques in robotics and control system.

Shouvik Dey received his PhD and M.Tech degree in Computer Science and Engineering from Jadavpur University (West Bengal, Kolkata, India) in 2012 and 2004 respectively and B.Tech in Computer Science and Engineering from Kalyani University (West Bengal, India) in 2002. He has IT experience of 9.5 Years in IBM, TCS and Cognizent. He is having teaching experience of more than 3 Years. Presently he is working as an Assistant professor in the Department of Computer Science and Engineering in National Institute of Technology, Nagaland, India. His research interests include data mining and distributed systems.