RESEARCH ARTICLE

A New Soft Computing Method for K-Harmonic Means Clustering

Wei-Chang Yeh[1] *, **Yunzhi Jiang**[2], **Yee-Fen Chen**[3], **Zhe Chen**[4]

1 Integration and Collaboration Laboratory, Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu City, Taiwan, **2** School of Automation, Guangdong Polytechnic Normal University, Guangdong, China, **3** Department of Industrial Engineering and Management, Hsiuping University of Science and Technology, Taichung City, Taiwan, **4** School of Information Engineeering, Chang'an University, Xi'an, China

* yeh@ieee.org

Abstract

The K-harmonic means clustering algorithm (KHM) is a new clustering method used to group data such that the sum of the harmonic averages of the distances between each entity and all cluster centroids is minimized. Because it is less sensitive to initialization than K-means (KM), many researchers have recently been attracted to studying KHM. In this study, the proposed iSSO-KHM is based on an improved simplified swarm optimization (iSSO) and integrates a variable neighborhood search (VNS) for KHM clustering. As evidence of the utility of the proposed iSSO-KHM, we present extensive computational results on eight benchmark problems. From the computational results, the comparison appears to support the superiority of the proposed iSSO-KHM over previously developed algorithms for all experiments in the literature.

Citation: Yeh W-C, Jiang Y, Chen Y-F, Chen Z (2016) A New Soft Computing Method for K-Harmonic Means Clustering. PLoS ONE 11(11): e0164754. doi:10.1371/journal.pone.0164754

Editor: Yong Deng, Southwest University, CHINA

Received: May 22, 2016

Accepted: October 2, 2016

Published: November 15, 2016

Data Availability Statement: All relevant data are within the paper.

Funding: This research was also supported in part by the National Science Council of Taiwan, R.O.C. under grant NSC101-2221- E-007-079- MY3 and NSC 102-2221-E-007-086-MY3. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

1. Introduction

Clustering is perhaps the most well-known technique in data mining to cluster data based on certain criteria. In past decades, clustering has attracted much attention, and it is increasingly becoming an important tool due to its wide and valuable applications in improving data analysis in various fields, such as the natural sciences, psychology, medicine, engineering, economics, marketing and other fields [1–28].

Clustering is an NP-hard problem with computational effort growing exponentially with the problem size [1–3]. There are two categories among all existing clustering algorithms: hierarchical clustering and partition clustering [3]. The former builds a hierarchy tree of data that successively merges similar clusters, while the latter begins with a random partition and refines it iteratively [3].

The most popular class of partition clustering is the centroid-based clustering algorithm. Among all clustering methods, with an extensive history dating back to 1972, K-means (KM) is one of the most well-known center-based partition clustering techniques [4–17]. KM is implemented by first randomly selecting *K* initial centroids and then trying to minimize heuristically the sum of the squares of distances, e.g., the Euclidean distance, Manhattan distance, and Mahalanobis distance, between each data point to the centroids [4–16].

As seen above, KM is relatively simple, even on large datasets. Hence, it has effective widespread applications in various real-life problems, such as market segmentation, classification analysis, artificial intelligence, machine learning, image processing, and machine vision [4–16]. Moreover, KM is implemented frequently as a preprocessing stage for other methodologies as a starting configuration.

However, KM is a heuristic algorithm and has two serious drawbacks [7–16]:

1. Its result depends on the initial random clusters, i.e., sensitivity to initial starting centroids;
2. It may be trapped in a local optimum; i.e., there is no guarantee that it will converge to the global optimum.

Therefore, the K-harmonic means (KHM) algorithm was proposed by Zhang [7] in 1999 to solve the problem of sensitivity to initial starting points. However, it still may be trapped by convergence to a local optimum. Hence, the main focus of KHM research has shifted to develop soft computing, such as the tabu K-harmonic means [9], simulated annealing based KHM [10], the particle swarm optimization (PSO) KHM (PSO-KHM) [11], the hybrid data clustering algorithms based on ant colony optimization and KHM [12], a variable neighborhood search (VNS) for KHM clustering [10], the multi-start local search for KHM clustering (MLS) [13], the gravitational search algorithm based KHM [14], the candidate groups search combined with K-harmonic mean (CGS-KHM) [15], the simplified swarm optimization based KHM (SSO-KHM) [16], the statistical feature extraction modeling KHM [29], the PSO hybrid with tabu search for KHM clustering [30], the firefly [31] and the enhanced firefly algorithm [32] for KHM clustering, the fish school search algorithm [33], and the genetic hybrid with gravitational search for KHM clustering [34], to avoid the local trap problem and reduce numerical difficulties.

Soft computing is able to help the traditional KHM methods escape from the local optimum trap and obtain better results [7–16]. However, the update mechanisms of these soft computing methods are either too tedious, which then requires extra computational efforts, or too weak in their local search, which requires more time for convergence [16]. Thus, there is always a need to have a better soft computing method for KHM clustering.

In this paper, a new algorithm, iSSO-KHM, is proposed to help the KHM escape from local optima by installing a new update mechanism into the SSO and integrating the KHM. The rest of the paper is organized as follows: Section 2 provides a description of the KHM and an overview of SSO. The novel one-variable difference update mechanism and the survival of the fittest policy, which are two cores in the proposed iSSO-KHM, are introduced in Section 3. Section 4 compares the proposed iSSO-KHM with three recently introduced KHM-based algorithms in eight benchmark datasets adopted from the UCI database to demonstrate the performance of the proposed iSSO-KHM. Finally, concluding remarks are summarized in Section 5.

2. Overview of SSO and KHM

The proposed iSSO-KHM is based on both SSO and KHM. Before discussing the proposed iSSO-KHM, how to solve the KHM clustering, basic SSO and KHM algorithms is introduced formally in this section.

2.1 The SSO

SSO is a new population-based soft computing method that was introduced originally by Yeh for discrete-type optimization problems [17] and has applications in two hot research topics in soft computing: swarm intelligence and evolutionary computing. From the applications in various optimization problems, SSO has demonstrated its simplicity, efficiency, and flexibility at exploring large and complex spaces [16–28].

Let Nsol be the number of solutions that are initialized randomly, K be the number of variables and the number of centroids, $\mathbf{c}_i = (c_{i,1}, c_{i,2}, \ldots, c_{i,K})$ be the ith solution inside the problem space with a fitness value $F(\mathbf{c}_i)$ determined by the fitness function F to be optimized, *pBest* $P_i = (p_{i,1}, p_{i,2}, \ldots, p_{i,K})$ be the best fitness function value of the ith solution with its own history, and *gBest* $P_{gBest} = (p_{gBest,1}, p_{gBest,2}, \ldots, p_{gBest,K})$ be the solution with the best fitness function value among all *pBest*s, where $i = 1, 2, \ldots,$ Nsol and $gBest \in \{1, 2, \ldots,$ Nsol$\}$.

Analogous to all other soft computing techniques, SSO searches for optimal solutions by updating generations. In every generation of SSO, each variable $c_{j,k}$ is updated according to the following simple step function after C_w, C_p, and C_g are given:

$$c_{i,k} = \begin{cases} c_{j,k} & \text{if } \rho_C \in [0, C_w) \\ p_{i,k} & \text{if } \rho_C \in [C_w, C_p) \\ p_{gBest,k} & \text{if } \rho_C \in [C_p, C_g) \\ x & \text{if } \rho_C \in [C_g,\ 1) \end{cases}. \tag{1}$$

where $j = 1, 2, \ldots,$ Nsol; $k = 1, 2, \ldots,$ K; C_w, $C_p - C_w$, C_g, and $1 - C_g$ are the predefined probabilities to determine whether $c_{j,k}$ will be updated to the same value (i.e., no change); $p_{j,k}$ in its *pBest*, $p_{gBest,k}$ of *gBest*, and regenerated to a new randomly generated feasible value [16–28].

Moving toward *pBest* is a local search; moving toward *gBest* is a global search. Moving toward a randomly generated feasible value is also a global search to maintain population diversity and enhance the capacity of escaping from a local optimum. Thus, each solution is a compromise among the current solution, *pBest*, *gBest*, and a random movement; this process combines local search and global search, yielding high search efficiency [16–28].

2.2 The KHM

KHM is similar to KM [7–16]. It is also a center-based partition clustering and randomly selects K initial centroids in the beginning. The major difference between KHM and KM is that KHM uses harmonic averages of the distances from each data point to the centers as components of its performance function. The detail of the KHM clustering algorithm is shown as follows [7–16]:

KHM PROCEDURE.

STEP K1. Select K initial centroids $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K$ randomly, where $\mathbf{c}_k$ is the centroid of the kth cluster; let F^* be a large number, and provide a tolerance ε.

STEP K2. Calculate fitness function:

$$F(\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_k) = \sum_{i=1}^{N} \frac{K}{\sum_{k=1}^{K} \frac{1}{\|X_i - \mathbf{c}_k\|^p}}, \tag{2}$$

where p is the p^{th} power of the Manhattan distance.

STEP K3. If $(F^* / F(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K) - 1 < \varepsilon)$, then halt and go to STEP K7; else, let $F^* = F(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K)$.

STEP K4. Calculate the membership of each data X_i to centroids $\mathbf{c}_k$ for $i = 1, 2, \ldots, N$ and $k = 1, 2, \ldots, K$ as below:

$$M(\mathbf{c}_k, X_i) = \frac{\|X_i - \mathbf{c}_k\|^{-p-2}}{\sum_{k=1}^{K} \|X_i - \mathbf{c}_k\|^{-p-2}}, \tag{3}$$

STEP K5. Calculate the weight of each data X_i for $i = 1, 2, \ldots, N$ as below:

$$W(X_i) = \frac{\sum_{k=1}^{K} \|X_i - \mathbf{c}_k\|^{-p-2}}{(\sum_{k=1}^{K} \|X_i - \mathbf{c}_j\|^{-p})^2}. \tag{4}$$

STEP K6. Calculate the new centroid $\mathbf{c}_k$ for $k = 1, 2, \ldots, K$ as below and go to STEP K2:

$$c_k = \frac{\sum_{i=1}^{N} M(\mathbf{c}_k,\ X_i) \cdot W(X_i) \cdot X_i}{\sum_{i=1}^{N} M(\mathbf{c}_k,\ X_i) \cdot W(X_i)} \tag{5}$$

STEP K7. Assign data point X_i to cluster k if $\mathrm{M}(\mathbf{c}_j, X_i) \leq \mathrm{M}(\mathbf{c}_k, X_i)$ for $j = 1, 2, \ldots, K$.

STEP K2 calculates the fitness function $F(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K)$ of KHM by summing up all harmonic averages of the distances between each data point and all centroids. STEP K3 defines the stopping criteria for KHM. In STEP K4, KHM employs each member function $M(\mathbf{c}_k, X_i)$ to measure the influence over the centroid $\mathbf{c}_k$ to data X_i. This member function determines which cluster each data point belongs to in STEP K7. STEP K5 assigns dynamic weight $W(X_i)$ to each data point such that the larger the weight is, the smaller the distance is to any centroid to avoid multiple centroids close together. STEP K6 updates the current centroids.

3. The Proposed iSSO-KHM

Based on the novel one-variable difference update mechanism and the policy of survival of the fittest, the proposed iSSO-KHM is able to find a good solution without needing to explore all possible combinations of solutions. These two parts, i.e., the novel one-variable difference update mechanism and the policy of survival of the fittest, are discussed in this section.

3.1 The one-variable difference update mechanism

Each soft computing method has its own generic update mechanism and numerous revised update mechanisms for different applications in various situations. In most soft computing, the update mechanism is only changed slightly. For example, the update mechanism of PSO is considered to be a vector-based update mechanism using the following two equations where c_1 and c_2 are two constants:

$$V_i^{t+1} = wV_i^t + c_1 \cdot \rho_1 \cdot (P_i^t - X_i^t) + c_2 \cdot \rho_2 \cdot (P_{gBest}^t - X_i^t) \tag{6}$$

$$X_i^{t+1} = X_i^t + V_i^{t+1}. \tag{7}$$

Note that all variables in the same solution share two random variables in PSO, i.e., ρ_1 and ρ_2 which are generated randomly from a uniform distribution within [0, 1] in Eq 6. In ABC, one variable for each solution is selected randomly for updating. The updated operators in traditional GA are either two variables via one-cut-point mutation, or up to half the number of variables changed via one-cut-point crossover. In the traditional SSO, however, all variables are updated simultaneously based on Eq 1.

To reduce the number of random values and to change solutions gradually without breaking the trend and stability in the convergent status, only one variable is updated in each solution for each iteration in the proposed iSSO-KHM. Another reason to adapted the one-variable update mechanism is due to the specific factor that the KHM is essentially insensitive to the initial conditions and only needs to refine its solution [7–16].

The update mechanism listed in Eq 1 is more suitable for this discrete data or type, and each variable of centroids is a floating point value in the KHM. Hence, the step function in Eq 1 is also revised for floating-point data in the novel one-variable difference update mechanism for the proposed iSSO-KHM as follows:

$$c_{j,k} = c_{j,k} + \rho_1 - \rho_2 \cdot \begin{cases} (c_{gBest,k} - c_{j,k}) & \text{if } \rho_C \in [0, C_g) \\ (c_{gBest,k} - c_{y,k}) & \text{if } \rho_C \in [C_g, C_w)\,, \\ (c_{x,k} - c_{j,k}) & \text{otherwise} \end{cases} \tag{8}$$

where ρ_1, ρ_2, and ρ_c are random numbers generated from the uniform distribution within [0,1]. Note that $C_g = .4$ and $C_w = .6$ in this study, the role of *pBest* is removed, and the comparison order is C_g first and then C_w in the step function of Eq 6, which is different from Eq 1.

For example, let $\mathbf{c}_3 = (1.3, 4.5, 6.7, 8.9)$ be the current solution, $\mathbf{c}_{gBest} = \mathbf{c}_6 = (2.7, 7.6, 5.4, 9.8)$ be the *gBest*, $\mathbf{c}_x = \mathbf{c}_5 = (2.3, 5.5, 7.7, 9.9)$ and $\mathbf{c}_y = \mathbf{c}_7 = (6.2, 8.5, 1.7, 4.9)$ be two randomly selected solutions, and the third variable (i.e., $c_{3,3}$) be selected randomly to update. Assume that $\rho_1 = 0.3$ and $\rho_2 = 0.6$ are generated randomly. Table 1 shows the newly updated $\mathbf{c}_3$ for three different cases resulting from three different values of ρ_c:

3.2 Survival-of-the-fittest policy

The policy of survival of the fittest, inspired by natural selection, is a strategy to select the most fits and eliminate unfits. In the traditional SSO, the updated solution must replace the old solution regardless of whether the updated solution is worse [17–28]. However, *gBest* is based on survival-of-the-fittest policy; i.e., only a solution that is better than the *gBest* can replace *gBest* [17–28].

Unlike SSO, the proposed one-variable difference update mechanism only updates one variable and places more emphasis on the local search. Additionally, KHM is less sensitive to the

Table 1. The new update c_3 for three different cases.

Case	ρ_c	original $c_{3,3}$	updated $c_{3,3}$	updated c_3
1	0.12	6.7	6.7+0.3·0.6·(5.4−6.7) = 6.466	(1.3, 4.5, **6.466**, 8.9)
2	0.45	6.7	6.7+0.3·0.6·(5.4−1.7) = 7.366	(1.3, 4.5, **7.366**, 8.9)
3	0.99	6.7	6.7+0.3·0.6·(7.7−1.7) = 7.780	(1.3, 4.5, **7.780**, 8.9)

doi:10.1371/journal.pone.0164754.t001

updated solutions. Hence, the survival-of-the-fittest policy applies to both *gBest* and all updated solutions to reduce the evolution time.

3.3 The complete pseudocode of the proposed iSSO-KHM

Like the existing related KHM algorithms, the KHM procedure discussed in section 2.2 to calculate the fitness of each solution is implemented in the iSSO-KHM and acts as a local search to further improve each updated solution heuristically. The steps of complete pseudocode of the proposed iSSO-KHM are described as follows.

iSSO-KHM PROCEDURE.

```
STEP 0. Generate $\mathbf{c}_j = (c_{j,1}, c_{j,2}, \ldots, c_{j,K})$ randomly, update $\mathbf{c}_j$, and calculate its fitness using the KHM procedure discussed in Section 2.2 for all $j = 1, 2, \ldots$, Nsol.
STEP 1. Let gen = 1 and find $gBest \in \{1, 2, \ldots, \text{Nsol}\}$ such that $F(\mathbf{c}_{gBest}) \le F(\mathbf{c}_j)$ for all $j = 1, 2, \ldots$, Nsol.
STEP 2. Let $j = 1$.
STEP 3. Select a variable (i.e., a centroid) randomly from $\mathbf{c}_j$, say $c_{j,k}$ where $k \in \{1, 2, \ldots, \text{K}\}$, and let $\mathbf{c}^* = \mathbf{c}_j$ and $F^* = F(\mathbf{c}_j)$.
STEP 4. Generate a random number $\rho_C$ from the uniform distribution between [0, 1]
STEP 5. If $\rho_C < C_g$, then let $x = gBest$, $y = j$, and go to STEP 8.
STEP 6. If $\rho_C < C_w$, then let $x = gBest$, select $y$ randomly from $\{1, 2, \ldots, K\}$, and go to STEP 8.
STEP 7. Select two integers $x$ and $y$ randomly from $\{1, 2, \ldots, K\}$.
STEP 8. Let $c_{j,k} = c_{j,k} + \rho_{[0,1]} \cdot \rho_{[0,1]} + (c_{x,k} - c_{y,k})$, and run the procedure KHM to update $\mathbf{c}_j$ and calculate its fitness.
STEP 9. If $F(\mathbf{c}_j) > F^*$, then let $\mathbf{c}_j = \mathbf{c}^*$ and $F(\mathbf{c}_j) = F^*$, and go to STEP 11.
STEP 10. If $F(\mathbf{c}_j) < F(\mathbf{c}_{gBest})$, then let $gBest = j$.
STEP 11. If the runtime is less than the predefined $T$, then go to STEP 2; otherwise, $\mathbf{c}_{gBest}$ is the final solution, and halt.
STEP 12. If $j <$ Nsol, let $j = j+1$, and go to STEP 3.
```

In the above, STEP 0 simply runs the KHM procedure for each randomly generated solution to calculate its fitness function and update the solution. STEP 1 finds the first *gBest* from these initial populations after using the KHM procedure. STEPs 2–12 implement the proposed one-variable difference update mechanism; STEPs 9 and 10 are based on the survival-of-the-fittest policy to decide whether to accept the updated solution or replace *gBest*. Note that the stopping criterion in STEP 11 is the runtime T, and T = 0.1, 0.3, and 0.5 CPU seconds in the experiments tested in Section 4.

4. Experimental Results

In this section, we present the computational results of the comparisons among the proposed algorithm and existing algorithms on eight benchmark datasets to test the performance of iSSO-KHM.

4.1 The Experimental Setting

To evaluate the efficiency and effectiveness (i.e., the solution quality) of the proposed iSSO-KHM, eight benchmarks adopted from UCI are tested: Abalone (denoted by A, 4177 records and seven features), Breast-Cancer-Wisconsin (denoted by B, 699 records and nine features), Car (denoted by C, 1728 records and six features), Glass (denoted by G, 214 records and nine features), Iris (denoted by I, 150 records and four features), Segmentation (denoted by S, 2310 records and 19 features), Wine (denoted by W, 178 records and 13 features), and Yeast (denoted by Y, 1484 records and eight features).

Moreover, iSSO-KHM is compared to four KHM-related soft computing algorithms: CGS_KHM, MLS_KHM, PSO_KHM, and SSO_KHM. Note that CGS_KHM has better performance than tabu search and VNS for the Iris, Glass and Wine datasets.

The programming language used was C++ with default options for all five algorithms: CGS_KHM (denoted by CGS), iSSO-KHM (denoted by iSSO), MLS_KHM (denoted by MLS), PSO_KHM (denoted by MLS), and SSO_KHM (denoted by SSO). All codes were run using a 64-bit Window 10 Operating System with Intel Core i7-5960X 3.00 GHz CPU and 16 GB of RAM.

In experiments, all values of K are set to three; the p^{th} power of the Manhattan distance is $p = 1.5$, 2.0, and 3.0; and the runtime limit is $T = 0.1$, 0.3, and 0.5 CPU seconds. For each test and algorithm, the number of solutions is 15, i.e., Nsol = 15, the number of independent runs is 55, and only the best 50 results are recorded to remove possible outliers; the stopping criteria are $T = 0.1$, 0.3, and 0.5 CPU seconds.

All required parameters for CGS, MLS, PSO, and SSO are taken directly from [15], [13], [11], and [20] for a fair comparison; two parameters, $C_g = 0.4$ and $C_w = 0.6$, are used in the proposed iSSO-KHM.

In all tables listed in S1 Appendix and the following two subsections, the notations F_{avg}, F_{min}, F_{max}, and F_{std} denote the average, minimal (the best), maximal (the worst) and standard deviation of the fitness values obtained from related algorithms. Additionally, the notations f_{avg}, f_{min}, f_{max} and f_{std} represent the number of F_{avg}, F_{min}, F_{max} and F_{std} that are the best among all algorithms under the same related conditions, e.g., p, T, and/or dataset.

To compare the efficiency of the update mechanism of the proposed iSSO, the average of the corresponding fitness calculation number (N_{avg}) and the number of best N_{avg} represented by n_{avg} are recorded. Note that for a fixed T, a higher N_{avg} means that the related update mechanism is more efficient and increases the search performance for finding an optimal solution.

To properly evaluate the clustering method, the $\text{F}_{\text{measure}}$ value is provided and the number of best $\text{F}_{\text{measure}}$ [35,36] is represented by f_{mea}. The $\text{F}_{\text{measure}}$ is one of the standard clustering validity measures based on the ideas of precision and recall from information retrieval [35,36]. Evidently, the bigger value of $\text{F}_{\text{measure}}$ is, the higher the quality of clustering is.

All experimental results are listed in S1 Appendix. S1 Appendix demonstrates that iSSO has achieved better solutions for each test problem with lower standard deviations and higher fitness computation numbers compared to the other methods.

4.2 General Observations for f_{avg}, f_{min}, f_{max}, f_{std}, and n_{avg}, and f_{mea}

All results in S1 Appendix are ranked and discussed in this subsection. Tables 2–5 summarize these ranking based on different T and p, T only, p only, and algorithm only, respectively. The letter next to the number denotes the related dataset, e.g., B2S denotes one best value in dataset B and two best value in dataset S.

From Table 2, iSSO has higher numbers in f_{avg}, f_{min}, f_{max}, f_{std}, n_{avg}, and f_{mea} than other methods for different setting of T and p. Hence, iSSO is more efficient, effective, and robust than other methods.

Table 3 summarizes the values of f_{avg}, f_{min}, f_{max}, f_{std}, n_{avg}, and f_{mea} for $T = 0.1$, 0.3, and 0.5 separately. We can observe that the longer runtime is, the better the solution quality obtained

Table 2. The number of f_{avg}, f_{min}, f_{max}, f_{std}, n_{avg}, and f_{mea}.

	T	**0.1**						**0.3**						**0.5**					
p	**Alg.**	**f_{avg}**	**f_{min}**	**f_{max}**	**f_{std}**	**n_{avg}**	**f_{mea}**	**f_{avg}**	**f_{min}**	**f_{max}**	**f_{std}**	**n_{avg}**	**f_{mea}**	**f_{avg}**	**f_{min}**	**f_{max}**	**f_{std}**	**n_{avg}**	**f_{mea}**
1.5	CGS	S	0	S	BS	0	0	S	0	S	S	0	W	0	0	0	0	0	0
	iSSO	**6**	**6**	**7**	**6**	**7**	**5**	**7**	**7**	**7**	**7**	**8**	**7**	**8**	**8**	**8**	**8**	**8**	**7**
	MLS	0	S	0	0	A	0	0	0	0	0	0	0	0	0	0	0	0	S
	PSO	0	0	0	0	0	W	0	0	0	0	0	0	0	0	0	0	0	0
	SSO	A	A	0	0	0	AS	0	A	0	0	0	0	0	0	0	0	0	0
2.0	CGS	S	S	S	BS	0	0	S	0	S	S	0	0	0	0	0	0	0	0
	iSSO	**7**	**6**	**7**	**6**	**7**	**6**	**6**	**8**	**6**	**6**	**8**	**5**	**7**	**8**	**7**	**7**	**8**	**5**
	MLS	0	0	0	0	A	W	B	0	B	B	0	W	B	0	B	B	0	0
	PSO	0	0	0	0	0	0	0	0	0	0	0	S	0	0	0	0	0	I
	SSO	0	A	0	0	0	A	0	0	0	0	0	A	0	0	0	0	0	AY
2.5	CGS	S	0	S	BS	0	0	0	0	S	S	0	0	0	0	0	0	0	0
	iSSO	**7**	**6**	**7**	**6**	**7**	**5**	**8**	**8**	**7**	**7**	**8**	**5**	**8**	**8**	**8**	**8**	**8**	**6**
	MLS	0	S	0	0	A	W	0	0	0	0	0	0	0	0	0	0	0	G
	PSO	0	0	0	0	0	IS	0	0	0	0	0	BW	0	0	0	0	0	0
	SSO	0	A	0	0	0	0	0	0	0	0	0	A	0	0	0	0	0	I

doi:10.1371/journal.pone.0164754.t002

from iSSO in Table 3. For example, f_{min} is increased from 18 to 23 for $T = 0.1$ to $T = 0.2$. PSO is the second best in f_{mea} for both $T = 0.1$ and 0.3; SSO is the second best in f_{min} for T = 0.1 and 0.2, and in f_{mea} for $T = 0.3$. Additionally, as seen from Table 3, iSSO tends to perform much better than other methods from time to time, e.g., there are six cases in which F_{min} are better than that of iSSO for $T = 0.1$ but none in which F_{min} is better than that of iSSO for $T = 0.3$.

Table 4 sums up the values of f_{avg}, f_{min}, f_{max}, f_{std}, n_{avg}, and f_{mea} for $p = 1.5$, 2.0, and 2.5 separately. It is evident that iSSO is still the best method compared to the others in all aspects. According to published results, other methods work more effectively when p = 2.0 [7–16]. However, given the results, iSSO still retains its performance, regardless of the value of p. For example, f_{min} is 21 for $p = 1.5$ and 22 for both $p = 2.0$ and 2.5. Interesting observations can still

Table 3. The values of f_{avg}, f_{min}, f_{max}, f_{std}, n_{avg}, and f_{mea} for $T = 0.1$, 0.3, and 0.5.

T	**Alg.**	**f_{avg}**	**f_{min}**	**f_{max}**	**f_{std}**	**n_{avg}**	**f_{mea}**
0.1	CGS	3 (S)	1 (S)	3 (S)	6 (3B,3S)	0	0
	iSSO	**20**	**18**	**21**	**18**	**21**	**16**
	MLS	0	2 (S)	0	0	3 (A)	2 (W)
	PSO	0	0	0	0	0	3 (I,S,W)
	SSO	1 (A)	3 (A)	0	0	0	2 (A,S)
0.3	CGS	2 (S)	0	3 (S)	3 (S)	0	1 (W)
	iSSO	**21**	**23**	**20**	**20**	**24**	**17**
	MLS	1 (B)	0	1 (B)	1 (B)	0	1 (W)
	PSO	0	0	0	0	0	3 (B,S,W)
	SSO	0	1 (A)	0	0	0	2 (A)
0.5	CGS	0	0	0	0	0	0
	iSSO	**23**	**24**	**23**	**23**	**24**	**18**
	MLS	1 (B)	0	1 (B)	1 (B)	0	2 (G,S)
	PSO	0	0	0	0	0	1 (I)
	SSO	0	0	0	0	0	3 (A,I,Y)

doi:10.1371/journal.pone.0164754.t003

Table 4. The values of f_{avg}, f_{min}, f_{max}, f_{std}, n_{avg}, and f_{mea} for p = 1.5, 2.0, and 2.5.

p	Alg.	f_{avg}	f_{min}	f_{max}	f_{std}	n_{avg}	f_{mea}
1.5	CGS	2 (S)	0	2 (S)	3 (B,2S)	0	1 (W)
	iSSO	**21**	**21**	**22**	**21**	**23**	**19**
	MLS	0	1 (S)	0	0	1 (A)	1 (S)
	PSO	0	0	0	0	0	1 (W)
	SSO	1 (A)	2 (A)	0	0	0	2 (A,S)
2	CGS	2 (S)	1 (S)	2 (S)	3 (B,2S)	0	0
	iSSO	**20**	**22**	**20**	**19**	**23**	**16**
	MLS	2 (B)	0	2 (B)	2 (B)	1 (A)	2 (W)
	PSO	0	0	0	0	0	2 (I,S)
	SSO	0	1 (A)	0	0	0	4 (3A,Y)
2.5	CGS	1 (S)	0	2 (S)	3 (B,2S)	0	0
	iSSO	**23**	**22**	**22**	**21**	**23**	**16**
	MLS	0	1 (S)	0	0	1 (A)	2 (G,W)
	PSO	0	0	0	0	0	4 (B,I,S,W)
	SSO	0	1 (A)	0	0	0	2 (A,I)

doi:10.1371/journal.pone.0164754.t004

be found, as observed in Table 3, where, in general, CGS yields better results for the S dataset, than other datasets. PSO and SSO follow on in performance with f_{mea} in $p = 2.0$ and $p = 2.5$, respectively.

Table 5 lists the overall values of f_{avg}, f_{min}, f_{max}, f_{std}, n_{avg}, and f_{mea} for CGS, iSSO, MLS, PSO, and SSO separately. In general, it seems that iSSO is only slightly more powerful within dataset A as $f_{min} = 4$ and $f_{mea} = 5$ for SSO and for within dataset S as $f_{min} = 2$ for MLS, $f_{mea} = 2$ for PSO, and $f_{avg} = 5$ and $f_{max} = f_{std} = 6$ for CGS. This is similar to what is observed in Tables 2 and 3. However, the number of best values for iSSO in all statistical indexes are still more than 6.2 times better compared to those of other methods. For example, for f_{std}, CGS produced nine best values (3 in B dataset and 6 in S dataset), whereas iSSO produced 64 best values. This trend is also found when iSSO is compared across all algorithms and thus demonstrates that iSSO outperforms the other algorithms in almost all aspects.

4.3 General Observations for F_{avg}, F_{min}, F_{max}, F_{std}, N_{avg}, and $F_{measure}$

In general, each result obtained using the proposed iSSO is better than those obtained using the other methods described S1 Appendix and Section 4.2. For an elaborate analysis, the top five values of F_{min} for each dataset under all settings of *T* and *p* are summarized in Table 6 and discussed in this subsection.

In Table 6, the proposed iSSO has the largest number (33) of results among the top five values, and SSO, PSO, and CGS have four, two, and one results among the top five values of F_{min}, respectively. Note that in most cases SSO yields better results than CGS and MLS, as seen in Table 6, but all of the values of f_{avg}, f_{min}, f_{max}, f_{std} are zero for SSO in Section 4.2.

Table 5. The values of f_{avg}, f_{min}, f_{max}, f_{std}, n_{avg}, and f_{mea} for algorithms.

Alg.	f_{avg}	f_{min}	f_{max}	f_{std}	n_{avg}	f_{mea}
CGS	5 (S)	0	6 (S)	9 (3B,6S)	0	1 (W)
iSSO	**64**	**65**	**64**	**61**	**69**	**51**
MLS	2 (B)	2 (S)	2 (B)	2 (B)	3 (A)	5 (G, 3W, S)
PSO	0	0	0	0	0	7 (B, 2I, 2S, 2W)
SSO	1 (A)	4 (A)	0	0	0	8 (5A, I, S, Y)

doi:10.1371/journal.pone.0164754.t005

Additionally, we can see that the top four F_{min} values are all obtained from the proposed iSSO for all datasets, except iSSO only has the top two F_{min} in both A and S datasets, of which SSO has the 3rd and 4th best F_{min}, and PSO has the 5th best F_{min}. It seems that the algorithm with the best F_{min} also has the best F_{avg}, F_{max}, F_{std}, and N_{avg} in all datasets. However, the algorithm with the best F_{min} does not guarantee its $F_{measure}$ is also the best, this is applicable to A, B, C, G, I and W datasets.

The following are some other observations for p, T, N_{avg}, and F_{std}:

1. p: There are 14, 11, and 15 top-five F_{min} values across all the eight data sets for $p = 1.5$, 2.0, and 2.5 in Table 6, respectively. This debunks published literatures [7–16] which indicate that $p = 2.0$ yields the best result. The above observation is also found in Table 4 of Section 4.2.

2. T: The $T = 0.1$, 0.3, and 0.5 have 15, 16, and 9 top-five F_{min} values across all the eight data sets. Among top-five F_{min} values for each dataset, the one with the largest T also has the best F_{min}, e.g., $T = .3$ in G, I and W datasets and $T = 0.5$ in the rest of the datasets. Hence, the above result agrees with the basic concept in soft computing: more runtime results in better solution quality.

3. N_{avg}: The order of the best N_{avg} for each dataset from large to small is 9494.02 (I) > 5099.84 (G) > 5095.64 (W) > 1685.48 (B) > 769.6 (Y) > 722.9 (C) > 412.40 (S) > 254.26 (A), where the letter inside parentheses is the related dataset. The above order exactly coincides with the order from large to small of the number of recorders in each dataset: A (4177) > S (2310) > C (1728) > Y (1484) > B (699) > G (214) > W (178) > I (150), except when 5099.84 (G) > 5095.64 (W) in N_{avg}. Hence, the smaller the dataset is, the shorter the runtime is and the larger number of fitness calculations is.

4. F_{std}: The order of the best F_{std} for each dataset from small to large is 2.04E-11 (I) < 4.70E-09 (W) < 7.59E-09 (G) < 1.12E-06 (B) < 8.33E-06 (C) < 9.49E-04 (Y) < 3.92E-03 (A) < 1.51E+00 (S), where the letter inside parentheses is the related dataset. The above order of datasets is similar to that of N_{avg} because the more fitness calculations are performed, the lower is standard deviation.

5. Conclusions

In this work, a new soft computing method called the iSSO-KHM is proposed to solve the KHM clustering problem. The proposed iSSO-KHM adapted the fundamental concepts in both the traditional SSO and KHM by adding the novel one-variable difference update mechanism to update solutions and the survival-of-the-fittest policy to decide whether to accept the new update solutions.

The computational experiments compare the proposed iSSO-KHM with CGS, MLS, PSO, and SSO on eight benchmark datasets: Abalone, Breast-Cancer-Wisconsin, Car, Glass, Iris, Segmentation, Wine, and Yeast with settings of $K = 3$; $p = 1.5$, 2.0, and 2.5; and $T = 0.1$, 0.3, and 0.5.

The experimental results show the superiority of iSSO-KHM over the other three algorithms for almost all eight benchmark datasets. Hence, iSSO-KHM can achieve a trade-off between exploration and exploitation to generate a good approximation in a limited computation time systematically, efficiently, effectively, and robustly.

However, from the experiments in Section 4, the improved F_{min} value does not mean that the $F_{measure}$ is also improved. Therefore, a potential area of exploration would be to include $F_{measure}$ in the fitness function to improve both values of F_{min} and $F_{measure}$. Another limitation of the proposed algorithm is that C_g and C_w in Eq 8 of the proposed update mechanism must

Table 6. The top five F_{min} for each dataset.

ID	*T*	*p*	Alg.	F_{avg}	F_{min}	F_{max}	F_{std}	N_{avg}	$F_{measure}$
A	0.5	2.5	iSSO	377.100	377.096	377.115	3.92E-03	254.26	57.70%
	0.3	2.5	iSSO	377.129	377.097	377.226	3.42E-02	152.28	58.64%
	0.3	2.5	SSO	380.591	377.116	403.228	5.83E+00	146.32	58.67%
	0.5	2.5	SSO	385.472	377.157	414.058	1.04E+01	241.32	57.69%
	0.5	2.5	PSO	418.457	377.634	426.647	1.96E+01	238.2	57.59%
B	0.5	1.5	iSSO	424.867	424.867	424.867	1.12E-06	1685.48	96.15%
	0.3	1.5	iSSO	424.867	424.867	424.867	1.71E-05	1023.24	96.12%
	0.1	2.5	iSSO	424.886	424.867	424.972	2.66E-02	356.56	96.16%
	0.1	2	iSSO	424.883	424.867	424.998	2.42E-02	357.28	96.14%
	0.1	1.5	iSSO	424.888	424.868	424.967	2.71E-02	341.78	96.19%
C	0.5	1.5	iSSO	7068.628	7068.628	7068.628	8.33E-06	722.9	39.28%
	0.3	1.5	iSSO	7068.630	7068.628	7068.632	1.08E-03	436.18	39.25%
	0.1	2.5	iSSO	7069.794	7069.146	7070.432	3.35E-01	150.64	39.33%
	0.1	2	iSSO	7070.008	7069.150	7070.889	4.04E-01	150.66	39.39%
	0.1	1.5	iSSO	7069.977	7069.189	7070.878	4.25E-01	144.18	39.23%
G	0.3	1.5	iSSO	1059.336	1059.336	1059.336	7.59E-09	5099.84	41.42%
	0.3	2	iSSO	1059.336	1059.336	1059.336	6.87E-09	5100.06	41.33%
	0.1	2.5	iSSO	1059.336	1059.336	1059.337	6.21E-06	1780.36	41.39%
	0.1	1.5	iSSO	1059.336	1059.336	1059.337	5.95E-06	1705.48	41.39%
	0.1	2	iSSO	1059.336	1059.336	1059.337	9.23E-06	1779.64	41.50%
I	0.3	1.5	iSSO	181.728	181.728	181.728	2.04E-11	9494.02	75.31%
	0.3	2	iSSO	181.728	181.728	181.728	1.83E-11	9495.64	75.36%
	0.1	1.5	iSSO	181.728	181.728	181.728	1.08E-07	3185.04	75.25%
	0.1	2	iSSO	181.728	181.728	181.728	7.83E-08	3313.24	75.41%
	0.1	2.5	iSSO	181.728	181.728	181.728	6.70E-08	3316.62	75.26%
S	0.5	1.5	iSSO	42852347.416	42852345.652	42852350.697	1.51E+00	412.40	53.18%
	0.5	2	iSSO	42852347.460	42852345.678	42852351.912	1.61E+00	412.80	53.10%
	0.3	2	iSSO	42852454.042	42852348.975	42852724.919	9.39E+01	248.00	53.19%
	0.3	1.5	iSSO	42852441.557	42852356.230	42852682.740	7.27E+01	247.90	53.07%
	0.3	2	CGS	42852405.328	42852368.882	42852456.610	2.33E+01	184.10	53.14%
W	0.3	1.5	iSSO	5388248.279	5388248.279	5388248.279	4.70E-09	5095.64	62.07%
	0.3	2	iSSO	5388248.279	5388248.279	5388248.279	4.70E-09	5095.72	62.22%
	0.1	2	iSSO	5388248.279	5388248.279	5388248.279	4.69E-08	1776.94	62.15%
	0.1	1.5	iSSO	5388248.279	5388248.279	5388248.279	1.21E-07	1704.06	62.20%
	0.1	2.5	iSSO	5388248.279	5388248.279	5388248.279	6.11E-08	1779.74	62.08%
Y	0.5	2.5	iSSO	72.833	72.833	72.836	9.49E-04	769.6	56.21%
	0.3	2.5	iSSO	72.837	72.833	72.857	5.41E-03	465.74	56.15%
	0.3	2.5	SSO	74.094	72.952	75.856	8.53E-01	445.44	56.05%
	0.5	2.5	SSO	73.942	73.038	75.630	6.96E-01	734.42	56.09%
	0.3	2.5	PSO	74.343	73.185	75.418	6.40E-01	399.8	55.84%

doi:10.1371/journal.pone.0164754.t006

be known in advance, this also brings up another practical problem that is to develop a parameter free idea in the proposed algorithm in the future.

As there are some recently proposed swarm-based clustering algorithms, it is necessary to have more comparisons about the proposed algorithm with other well-known swarm-based clustering algorithms in the future. In Section 4, "Experimental results", the choice of the parameter *K* is fixed to 3. The proposed approach will also compare with the other versions of KHM for different values of *K* (like the case of *p* and *T* parameters).

Supporting Information

S1 Appendix.
(DOCX)

Acknowledgments

I wish to thank the anonymous editor and the reviewers for their constructive comments and recommendations, which have significantly improved the presentation of this paper. This research was supported in part by the National Science Council of Taiwan, R.O.C. under grant NSC101-2221- E-007-079- MY3 and NSC 102-2221-E-007-086-MY3.

Author Contributions

Conceptualization: W-CY.

Data curation: W-CY Y-FC.

Formal analysis: W-CY.

Funding acquisition: W-CY.

Investigation: W-CY.

Methodology: W-CY YJ.

Project administration: W-CY YJ.

Resources: W-CY ZC.

Software: W-CY Y-FC.

Supervision: W-CY YJ.

Validation: W-CY ZC.

Visualization: W-CY.

Writing – original draft: W-CY.

Writing – review & editing: W-CY.

References

1. Anderberg M.R., Cluster Analysis for Application, Academic Press, New York, 1973.
2. Mirkin B., Clustering for Data Mining: A Data Recovery Approach, Taylor and Francis group and FL, 2005.
3. Jain A.K., Murty M.N., Flynn P. J., Data clustering: A review, ACM Computational Survey 31(3) (1999) 264–323.
4. Steinhaus H., Sur la division des corps matériels en parties. Bull. Acad. Polon. Sci., 4 (1957) 801–804. MR 0090073. Zbl 0079.16403.
5. Forgy E.W., Cluster analysis of multivariate data: efficiency versus interpretability of classifications, Biometrics 21 (3) (1965) 768–769.
6. https://en.wikipedia.org/wiki/K-means_clustering.
7. B. Zhang, M. Hsu, U. Dayal, K-harmonic means–a data clustering algorithm, Technical Report HPL-1999-124, Hewlett–Packard Laboratories, 1999.

8. B. Zhang, Generalized k-harmonic means–boosting in unsupervised learning, Technical Report HPL-2000-137, Hewlett–Packard Laboratories, 2000.

9. Gungor Z., Unler A., K-harmonic means data clustering with tabu-search method, Applied Mathematical Modelling 32 (2008) 1115–1125.

10. Gungor Z., Unler A., K-harmonic means data clustering with simulated annealing heuristic, Applied Mathematics and Computation 184 (2007) 199–209.

11. Yang F., Sun T., Zhang C., An efficient hybrid data clustering method based on K-harmonic means and Particle Swarm Optimization, Expert Systems with Applications 36 (2009) 9847–9852.

12. Jiang H., Yi S., Li J., Yang F., Hu X., Ant clustering algorithm with K-harmonic means clustering, Expert Systems with Applications 37 (2010) 8679–8684.

13. Alguwaizani A., Hansen P., Mladenovic N., Ngai E., Variable neighborhood search for harmonic means clustering, Applied Mathematical Modelling 35 (2011) 2688–2694.

14. Yin M., Hu Y., Yang F., Li X., Gu W., A novel hybrid K-harmonic means and gravitational search algorithm approach for clustering, Expert Systems with Applications 38 (2011) 9319–9324.

15. Hung C.H., Chiou H.M., Yang W.N., Candidate groups search for K-harmonic means data clustering, Applied Mathematical Modelling 37 (2013) 10123–10128.

16. Yeh W. C., Lai C. M., Chang K. H., A novel hybrid clustering approach based on K-harmonic means using robust design, Neurocomputing 173 (2016), 1720–1732.

17. W. C. Yeh, Study on quickest path networks with dependent components and apply to RAP, Report, NSC 97-2221-E-007-099-MY3, 2008–2011.

18. Yeh W. C., A two-stage discrete particle swarm optimization for the problem of multiple multi-level redundancy allocation in series systems, Expert Systems with Applications 36 (2009) 9192–9200.

19. Yeh W., Chang W. and Chung Y., A new hybrid approach for mining breast cancer pattern us-ing discrete particle swarm optimization and statistical method, Expert Systems with Applications 36 (2009) 8204–8211.

20. Yeh W. C., Simplified Swarm Optimization in Disassembly Sequencing Problems with Learning Effects, Computers & Operations Research 39 (2012) 2168–2177.

21. Yeh W.C., Novel Swarm Optimization for Mining Classification Rules on Thyroid Gland Data, Information Sciences 197 (2012) 65–76.

22. Chung Y. Y., Wahid N., A hybrid network intrusion detection system using simplified swarm optimization (SSO), Applied Soft Computing, 12 (2012) 3014–3022.

23. Yeh W. C., New Parameter-Free Simplified Swarm Optimization for Artificial Neural Network Training and Its Application in the Prediction of Time Series, IEEE Transactions on Neural Networks and Learning Systems 24 (2013) 661–665. doi: 10.1109/TNNLS.2012.2232678 PMID: 24808385

24. Azizipanah-Abarghooee R., A new hybrid bacterial foraging and simplified swarm optimization algorithm for practical optimal dynamic load dispatch. International Journal of Electrical Power & Energy Systems, 49 (2013) 414–429.

25. Yeh W. C., Orthogonal simplified swarm optimization for the series–parallel redundancy allocation problem with a mix of components, Knowledge-Based Systems 64 (2014) 1–12.

26. Huang C.L., A particle-based simplified swarm optimization algorithm for reliability redundancy allocation problems, Reliability Engineering & System Safety 142 (2015) 221–230.

27. Lee J. H., Yeh W. C., Chuang M. C., Web page classification based on a simplified swarm optimization, Applied Mathematics and Computation 270 (2015) 13–24.

28. Yeh W.C., An improved simplified swarm optimization, Knowledge-Based Systems 82 (2015) 60–69.

29. M. W. Ayech, D. Ziou, Terahertz image segmentation based on k-harmonic-means clustering and statistical feature extraction modeling, in: International Conference Pattern Recognition, IEEE, Tsukuba, Japan, 2012, 222–225.

30. Aghdasi T., Vahidi J., Motameni H. and Inallou M.M.. K-harmonic means Data Clustering using Combination of Particle Swarm Optimization and Tabu Search, International Journal of Mechatronics, Electrical and Computer Technology 4(11) (2014) 485–501.

31. Abshouri A.A., Bakhtiary A., “A new clustering method based on firefly and KHM”, Journal of Communication and Computer 9(4) (2012) 387–39.

32. Zhou Z., Zhu S., Zhang D. A Novel K-harmonic Means Clustering Based on Enhanced Firefly Algorithm, Intelligence Science and Big Data Engineering. Big Data and Machine Learning Technique, Lecture Notes in Computer Science 9243 (2015) 140–149.

33. Serapiao A.B.S., Correa G.S., Goncalves F.B., Carvalho V.O., Combining K-Means and K-Harmonic with Fish School Search Algorithm for data clustering task on graphics processing units, Applied Soft Computing 41 (2016) 290–304.

34. Thakare A. D., Dhote C. A., Hanchate R. S., New Genetic Gravitational Search Approach for Data Clustering using K-Harmonic Means, International Journal of Computer Applications 99(13) (2014) 5–8.

35. Handl J., Knowles J., and Dorigo M., On the performance of ant-based clustering. Design and Application of Hybrid Intelligent Systems, Frontiers in Artificial Intelligence and Applications 104 (2003) 204–213.

36. A. Dalli, Adaptation of the F-measure to cluster-based Lexicon quality evaluation. In EACL 2003. Budapest.