Adv Syst Sci Appl 2018; **02**; 63-83
Published online at http://ijassa.ipu.ru/index.php/ijassa/article/view/576

WGW: A Hybrid Approach Based on Whale and Grey Wolf Optimization Algorithms for Requirements Prioritization

Amjad Hudaib[1], Raja Masadeh[2], Abdullah Alzaqebah[2]

[1]*Computer Information systems Department, the University of Jordan, Amman, Jordan*

E-mail: ahudaib@ju.edu.jo

[2]*Computer Science Department, the World Islamic Sciences and Education University, Amman, Jordan*

E-mail: raja.masadeh@wise.edu.jo , abdullah.zaqebah@wise.edu.jo

Abstract: Requirement engineering is the base phase of any software project, since this phase is concerned about requirements identification, processing and manipulation. The main source of these requirements is the project stakeholders with considering the project constraints and limitation. Number of requirement is varying for each project, so the requirements prioritization term comes for prioritizing the order of execution for software requirements according to the stakeholder's opinions and decisions. Various proposed optimization algorithms are employed to solve optimization problems; recently whale optimization (WO) algorithm is proposed in 2016 by Mirjalili which mimics the main characteristic of humpback whales which is the foraging method that is called bubble-net technique. On the other hand Grey wolf optimization (GWO) algorithm was proposed in 2014 in order to solve optimization problems by imitating the grey wolves hunting behavior. In this paper, a Hybrid approach based on Whale and Grey wolf optimization algorithms (WGW) is proposed by combining the advantages of each algorithm in order to prioritize the software requirements. Moreover, the data set that used in this paper is RALIC which a real software project's requirements is in order to evaluate the proposed method. Thus, the proposed method shows 91% accuracy of requirements prioritization comparing with RALIC data sat.

Keywords: Requirement prioritizations (RP), Whale Optimization Algorithm (WOA), Grey wolf optimization (GWO), Replacement Access, Library and ID Card project (RALIC), RP-WOA.

I. INTRODUCTION

Requirement Engineering (RE) in one of the most significant branch in the domain of Software Engineering. In addition, it is considered as the most important phase in Software Development Life Cycle [1]. This phase contains identification and elicitation of requirements, analysis and requirements validation and documentation. In almost software projects, there are restrictions on development process like budget and time to market production, this leads to deliver the software projects in consecutive releases, hence large projects have more than one of stakeholders that makes a difficulty in decision making about what release should be developed firstly. This difficulty contributes the software engineers to prioritize the requirements in efficient way to make the right decision about project's delivery and development. [1, 2, 3]

Thus, requirement prioritization (RP) is the most important section of RE that comes under the requirement analysis stage. RP is considered as one of the most significant activities in the process to construct software project and deliver good system as the customer need. In case the project has strict execution plan, insufficient resources and the expectations of the consumers are so high, then it must publish the most important characteristics as early as possible. Thus, this reason leads the requirements to be prioritized.

Requirement prioritization process increases the stakeholder's involvement by including them in deciding what requirements should be included in the software with its importance and effects on development process, and this involvement aids the stakeholders to really understand the restrictions and limitations on project's resources, and negotiate the conflicts between viewpoints that really effect in software development, and these conflicts come from different objectives and roles of stakeholders [43]. Thus, in projects that have large number of requirements, the prioritization become a base of software success or failure according to the project's constraints and limitations, these participation and involvement aims to prioritize the requirement according to their importance in efficient manner and useful order of execution [1, 2, 4]. According to that, the clustering technique could be used in order to classify the requirements based on their importance.

Data clustering is one of the most important functions in data mining, which has attracted many authors, researchers and experts recently. Clustering is an unsupervised ranking of types in groups [5]. The major purpose of data clustering depends on the notion and the idea of grouping the objects into groups that based on the similarity and dissimilarity between these objects [6, 7].

In general, clustering is categorized into two main classifications; the hard clustering and the soft clustering. In hard clustering, the data objects only belong to one cluster, also in the soft clustering; all the individual data objects belong to the individual classes to some range [6, 8]. The major goal of data clustering is which it optimally sorting all N data into K clusters in the opinion that the whole unseen types in the data are visible [9, 10]. Thus, each cluster contains the similarity data objects, as well as, the clusters are various from others. One of the most significant methods that are employed in the data exploration process, neural computing, image segmentation and other engineering is cluster analysis [11]. For the time being, many clustering techniques are proposed by researchers which categorized into model-based clustering algorithm, hierarchical clustering algorithm, partition-based clustering algorithm, grid-based clustering algorithm and density-based clustering algorithm [9, 13]. The data that are divided into K clusters utilizing the Euclidean distance as a measure in the partition-based clustering algorithms, as well the tree of groups that are created in the hierarchical clustering algorithms.

Recently, many researchers have proposed much of meta-heuristics and heuristic algorithms in order to solve the issues which happen as a result of complicated datasets. However, the most techniques that proposed in order to resolve the issue of the optimization relies on the meta-heuristic algorithms [11, 40-42]. The meta-heuristic algorithms main goal is to define the optimal solutions for fulfilling data cluster and reduces the issue of the local minima [14, 15]. The latest meta-heuristic optimizations algorithms are Grey Wolf Optimization (GWO) algorithm and Whale Optimization (WO) algorithm.

Grey Wolf Optimization (GWO) algorithm is proposed by Mirjalili et al in 2014 [12]. This mimics the hunting behavior of grey wolves in nature. Grey wolves are one of the well-known predators in the nature. Usually, they live in pack within group size is 5 to 12 on average. These wolves have robust rules in social dominant hierarchy. According to [12] grey wolves include alpha, beta, omega and delta wolves. Alpha wolf represents the leader of the pack and it is responsible for making decision about hunting and other activities. While the beta wolf helps the higher level to make decision. Omega wolves are responsible to submit any information to the highest levels. All other wolves in the pack are called delta.

Whale optimization (WO) algorithm is recently suggested by Mirjalili and Lewis in 2016 [13]. Where its main objective is to define the global optimal solution for any given optimization problem. The major distinction between this algorithm and other algorithms is the principle that develops the candidate solutions in each iteration of optimization. In addition to that, bubble- net feeding process represents the hunting process in humpback whales in order to find and attack the prey.

The reset of this paper is organized as follows: section II contains backgrounds of requirement prioritization, GWO algorithm and WO algorithm in details. While section III describes the proposed WGW and how it works. Section IV outlines the suggested algorithm "RP-WGW". The experimental results are presented in section V. Finally, section VI draws the conclusion and future work.

II.BACKGROUND

A.REQUIREMENTS PRIORITIZATION

The meaning of requirements prioritization is seen from several angles. Summerville defined the requirements prioritization as one of the most significant task for decision makers [16]. While, Firesmith defines it as the major process in software engineering as it gives perfect implementation order of the requirements in order to planning software versions and supplying desirable functionality as early as possible or the process to define the priority of the requirements to stakeholders based on the requirements importance [17]. Therefore, we can conclude that the requirements prioritization denotes the prioritization by importance or by implementation.

Implementing the most significant operations that leads to get incremental feedback from the customer, set schedule, solve mistakes and resolve any misunderstand between the customer and the corporation in premature phases that lead up to customer contentment. Moreover, it is valuable by eliminating needless requirements which may be inefficiently costly and choosing the most suitable requirements for each version; which leads to assist in future planning, reduce the risk of cancellation, evaluate the benefits, prioritize the investments and determine the financial effect with regards to the implementation of each requirement [18].

Requirement prioritization is the most significant and critical portion of requirements analysis due to the restrictions in project resources. In other words, it is so hard to implement the whole requirements simultaneously due to the restrictions in resources whence of schedule, staff and budget. Moreover, to improve some projects may require many months or often several years, wherefore it is important to determine the requirement that should be implemented at the beginning. Furthermore, budget plays an important factor, especially when transaction with requirement prioritization process because budget is considered as small activity with regards to requirement engineering compared to other activities in software engineering. Lastly, as mentioned before concludes that requirements have various levels with regards to their importance and it is complicate to determine which one is the most important.

As mentioned above, the project stakeholders are the base of the prioritization process with respect to business and regulations factors because they have various viewpoints and each one must determine the highest priority for requirements in order to impose stakeholders to clearly gather all the relative importance requirements that guides to raise the communication between stakeholders, supplies a reasonable base for requirement negotiation and enables engineering to schedule the development activities in reasonably.

B.GREY WOLF OPTIMIZATION

Grey Wolf optimization algorithm (GWO) is one of the latest bio-inspired optimization techniques that proposed by Mirjalili et al in 2014 [12]. Which imitate the hunting behavior of grey wolves in nature. Whereas the major purpose of GWO algorithm is determining the optimal for a given problem by using a population of search agents.

These wolves usually live in groups; their members between five and twelve. The major difference between GWO algorithm and the other optimization algorithms is the social dominant hierarchy which develops the candidate solution in each iteration of optimization. In facts, the GWO simulates the foraging behavior of the wolves in finding and attacking the victims [12, 19]. The social hierarchy of the wolves pack is shown in Fig.1 [12].

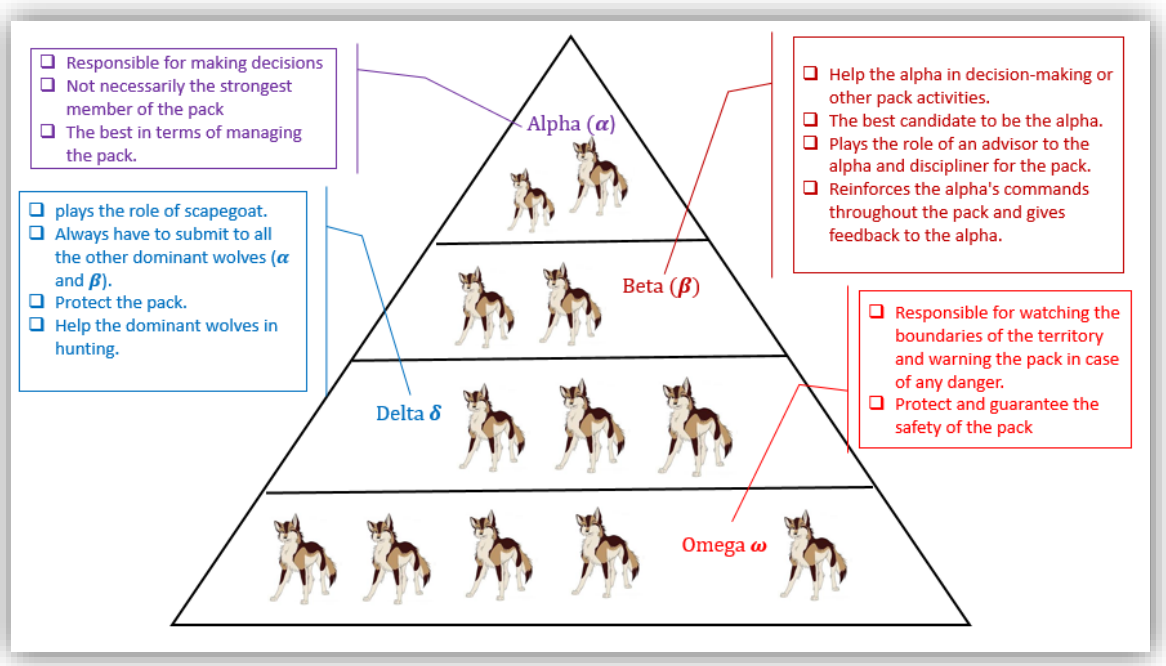

Fig.1. Hierarchy of grey wolf.

Alpha displays the leader which is the best candidate solution. In addition to that, alpha are dominant wolves which followed by the other wolves. While beta represents the second candidate solution which helping alpha in decision making and represent bridges between the leader and the rest of the pack that plays the lowest levels. Delta displays the third candidate solution that responsible to submit information to the two higher levels (alpha and beta). While omega's displays the rest of solutions. Moreover, its responsible to submit information to the three higher levels.

In fact, hunting mechanism contains three steps: tracking, encircling and attacking the prey. Thus, GWO represents the hunting technique of the grey wolves mathematical that is used to resolve complicated optimization problem. Thus, the optimal solution of the given problem is considered as victims.

The three top levels movement simulates the victim encirclement by grey wolves, which is the following formula that is suggested in this regard [12].

$$\vec{D} = \left|\vec{C}.\vec{X}_p(t) - \vec{X}(t)\right| \tag{1}$$

Where *t* displays the current iteration, X_p indicates to the prey position vector, *X* represents the grey wolf position and *C* is a coefficient vector. Thus, the result of vector D is used to move the particular element toward or away from the area that the best solution is located which represents the prey by using the following equation [12]:

$$\vec{X}(t+1) = \left|\vec{X}_p(t) - \vec{A}\,.\vec{D}\right| \text{, with } \vec{A} = 2\vec{a}.\vec{r}_1 - \vec{a} \tag{2}$$

Where r_1 is selected randomly in [0, 1] and a is minimized from 2 to 0 through predetermined number of iterations. In case |A| < 1, this matches to the exploitation behavior and simulates the behavior of attacking the prey. Otherwise, if |A|> 1, this matches exploration behavior and imitates the wolf spacing from the victim. The suggested values for A are in [-2, 2]. Thus, three higher levels α, β and δ will be computed using the following mathematical expressions [12].

$$\vec{D}_{\alpha} = \left|\vec{C}_1.\vec{X}_{\alpha} - \vec{X}\right| \text{ With } \vec{X}_1 = \vec{X}_{\alpha} - \vec{A}_{\alpha}.\left(\vec{D}_{\alpha}\right) \quad (3)$$

$$\vec{D}_{\beta} = \left|\vec{C}_2.\vec{X}_{\beta} - \vec{X}\right| \text{ With } \vec{X}_2 = \vec{X}_{\beta} - \vec{A}_{\beta}.\left(\vec{D}_{\beta}\right) \quad (4)$$

$$\vec{D}_{\delta} = \left|\vec{C}_3.\vec{X}_{\delta} - \vec{X}\right| \text{ With } \vec{X}_3 = \vec{X}_{\delta} - \vec{A}_{\delta}.\left(\vec{D}_{\delta}\right) \quad (5)$$

In order to mathematically imitative the hunting process of grey wolf, assume that α, β and δ have enough knowledge about the possible position of the victim. Moreover, the first three best solutions that gained are saved and force the other agents to update their locations according to the best agents α, β and δ. This behavior is represented mathematical by the following expression [12], and the pseudocode of the GWO is shown in Algorithm 1 [12].

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \quad (6)$$

Initialize the grey wolf population X_i (i = 1, 2, ..., n)
Initialize a, A, and C
Calculate the fitness of each search agent
X_α=the best search agent
X_β=the second best search agent
X_δ=the third best search agent
***while** (t < Max number of iterations)*
  ***for** each search agent*
    Update the position of the current search agent
  end for
  Update a, A, and C
  Calculate the fitness of all search agents
  Update X_α, X_β, and X_δ
  t=t+1
end while
return X_α

Algorithm 1: Pseudocodes of GWO [12]

C.WHALE OPTIMIZATION

Whale optimization algorithm (WOA) is a recently suggested randomly optimization algorithm by Mirjalili and Lewis in 2016 [13]. This algorithm purposes to determine the global optimum for the problem by using a population of search agents (whales). The search process is the first step begins with generating a collection of candidate solutions that are selected randomly for the given problem. Then, it ameliorates this collection during many numbers of iterations until the satisfaction of an end condition. In fact, Whales imitative private hunting technique that was called bubble-net feeding method as shown in Fig. 2 [13].

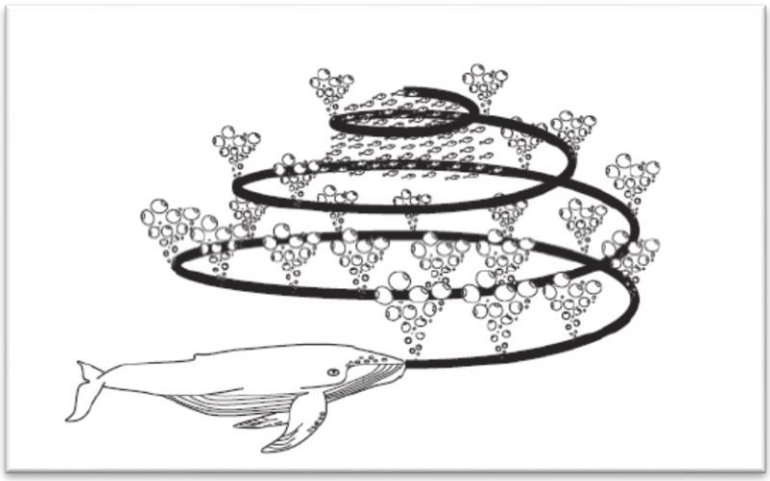

Fig. 2. Bubble-net hunting behavior [13]

It is obvious from Fig. 2 that a humpback whale generates ambush with shifting in a spiral route around the victims, then generates bubbles all the way ahead. Thus, this search process is the major inspiration of the whale optimization algorithm. In addition to that, the encircling method is another simulated technique in WOA. Whereas the humpback whales surrounding around the victims in order to begin hunting them using the foraging mechanism which is called the bubble- net technique. This behavior is represented mathematical by the following equations [13]:

$$\vec{X}(t+1) = \begin{cases} \vec{X}*(t) - \vec{A}.\vec{D} & if\ p < 0.5 \\ D'.e^{bl}.Cos\,(2\,\pi\,l) + \vec{X}*(t) & if\ p \geq 0.5 \end{cases} \quad (7)$$

Where p is a random number in [0, 1], b is a constant for determining the shape of the logarithmic spiral, and l indicates to a random number in [−1, 1], t presents the current iteration, and $D^{'} = |X^{*}(t) - X(t)|$ which mentions the distance between the ith whale and the victim.

In other words, the first phase is presented in this equation is the foraging mechanism that mimics the encircling technique, whereas the second phase simulates the bubble-net mechanism. The variable p exchanges between these two phases with similar probability. The potential cases those using these equations are shown in Fig.3 [13].

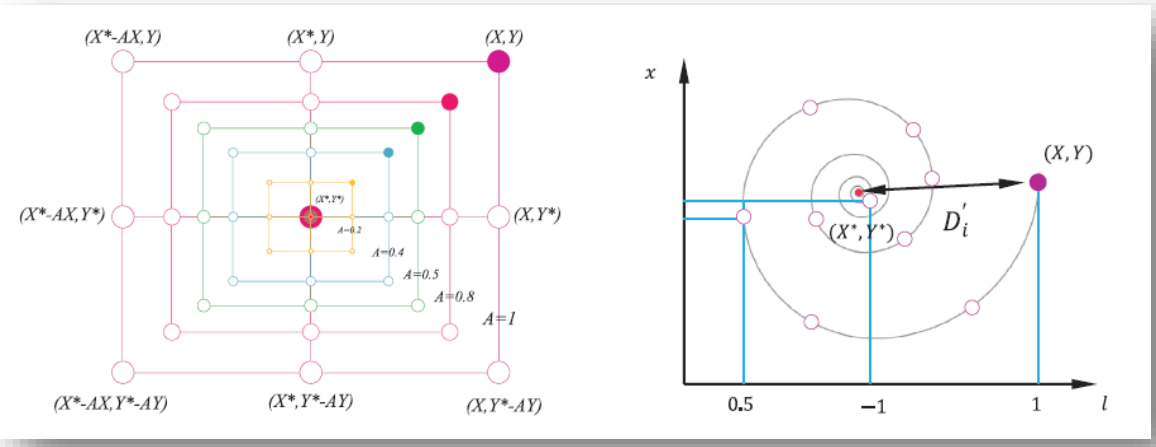

Fig.3. Mathematical models for prey encircling and bubble-net hunting [13]

The main two phases of the optimization algorithm by using population based algorithms are exploration and exploitation phases. Whereas both of them ensured in WOA by adaptively setting a and c in the major equation.

In case, a problem is given, the WOA begins optimizing this problem by generating collection of random solutions. In each iteration, the search agents update their location depends on the randomly chosen search agent or the best search agents that will gain so far. To assure the exploration phase, the other agents update their positions based on the best solution that represents the pivot point when |A|>1. In other case, when |A|<1 the best solution plays another role with the pivot point. The pseudocode of the WOA is shown in Algorithm 2 [13].

```
Initialize the whales population X_i (i = 1, 2, 3, ..., n)
Initialize a, A, and C
Calculate the fitness of each search agent
X* = the best search agent
procedure WOA(Population, a, A, C, MaxIter, .. )
    t = 1
    while t ≤ MaxIter do
        for each search agent do
            if |A| ≤ 1 then
                Update the position of the current search
                agent by the equation 2.6
            else if |A| ≥ 1 then
                Select a random search agent X_rand
                Update the position of the current agent
                by the equation 2.8
            end if
        end for
        Update a, A, and C
        Update X* if there is a better solution
        t = t + 1
    end while
    return X*
end procedure
```

Algorithm 2: Pseudocodes of WOA [13]

III.RELATED WORK

As mentioned before, the requirements prioritization phase is an operation that presenting the priority of a requirement over other requirements. Moreover, it is the most motivating field in the requirement engineering domain [21]. Many techniques are proposed in order to give the precedence to a software requirement over other requirements at the same project. Fig.4 shows the classification of requirements prioritization techniques according to [22].

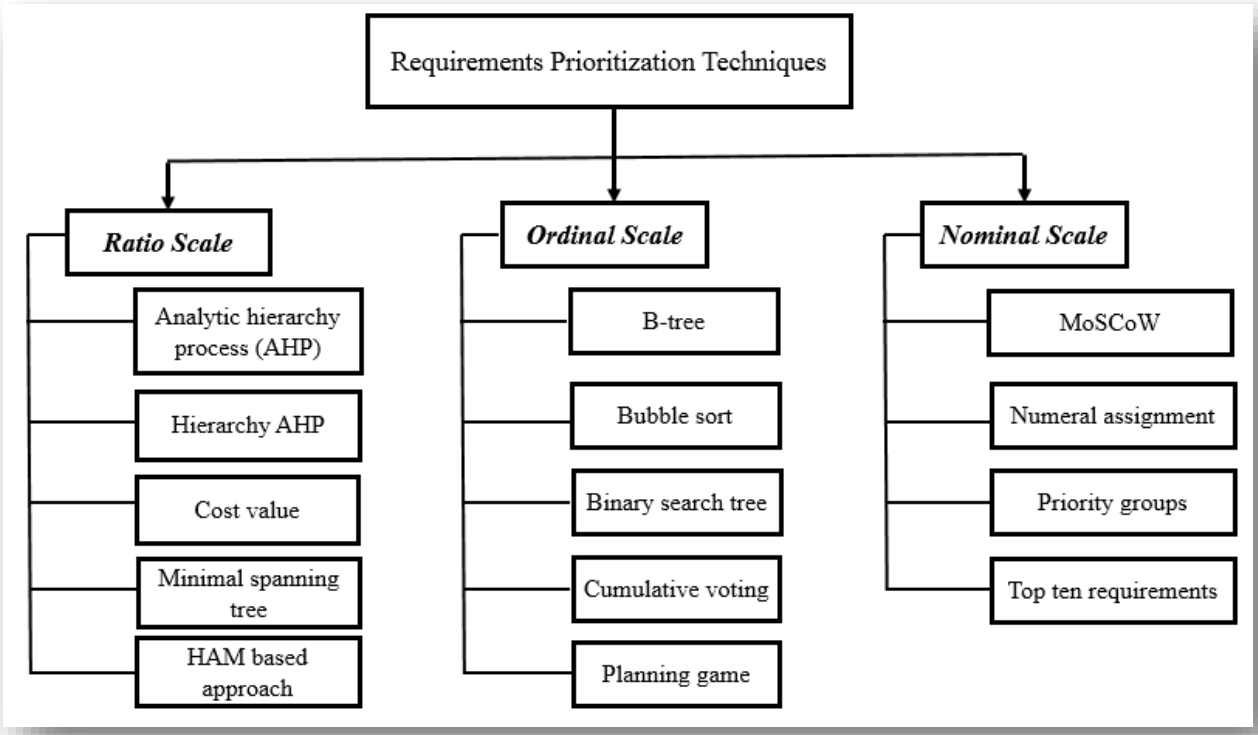

Fig.4. Classification of requirement prioritization techniques [22].

As shown in Fig.4 the requirement prioritization techniques are classified as ordinal scale, ordinal scale and ratio scale in terms of powerful prioritization. The most powerful prioritization is ratio scale that is giving how much more significant a requirement than the others requirements [42, 44]. While the least powerful prioritization is the ordinal scale which is responsible to give ranked list of requirements and giving which requirement is more important than others requirements but without giving how much more significant.

The Analytic Hierarchy Process (AHP) is the most popular and traditional technique that is joined ratio scale class. Thus, it is mentioned by large number of studies [23-28]. Furthermore, it is considered as a systematic decision making technique which has been performed for precedence of requirements for a particular project [29, 30]. It compares all potential pairs of requirements to order them and define which has higher priority. AHP is not suitable for a project that has large number of requirements [31, 32]. Thus, many researchers have attempted to minimize the number of comparisons such as [33, 34] and different techniques are presented in order to minimize the number of comparisons by approximately 75% such as [35].

The most precise level software requirements are placed at the base of the hierarchy while the most complicated requirements are placed on the top of the hierarchy [23]. Hierarchy AHP is another type of requirements prioritization technique that prefers the requirements that are presented at the same level.

Cumulative Voting is called the 100-Dollar Test which is a straightforward technique; where 100 imaginary units are given to the stakeholders in order to divide among the given software requirements [25]. The finding of the prioritization is given on a ratio scale.

Numerical assignment is considered as one of the most popular mechanism that is proposed in [24, 27, 47]. This technique depends on clustering requirements into various priority classes. However, the number of classes are vary and there are three classes are very popular [25, 28]. Thus, in this technique it is significant that each class displays something which the stakeholder can communicate to such as optional, critical, etc. While MoScoW (Museum of

Soviet Calculators on the Web) is considered as a type of numerical assignment mechanism that depicted in [36, 37]. Moreover, this technique has four priority classes; MUST have, COULD have, SHOULD have and WONT have. MUST have means the requirements that joined this class must be implemented at the first then goes to version, While COULD have means the requirement that joined this class exist then it will be great for the software product. SHOULD have means the requirements that present in this class are implemented then will be great for the software product and finally WONT have means that requirements join this cluster cannot be implemented in the present iteration as other requirements that are of low precedence.

Bubble sort mechanism is used to rank the element such as requirements. In this technique, two requirements are taken to compare with each other. In case the requirement is not in series then exchange between them and then compare it with another requirement and continue until get ranked list of requirements descending (from higher to lower) as used in these studies [23, 37].

Binary search tree technique is another type that is used for ranking which suggested by [37]. Furthermore, this technique was presented at the first time by [23] for requirement prioritization. Each node in this technique indicates to a software requirement where all requirements that located in left side of the tree are of lower priority comparing with other nodes while the other requirements that are located at the right side of the tree are of higher priority. However, at the first one requirement is selected to be the root node then will compare with unsorted requirement. In case this requirement is lower priority than the root, it searches the left side of the tree. Otherwise, it searches the right side of the tree. The operation is repeated until get sorted tree.

In top ten requirements technique, the stakeholders select top ten requirements in terms of their importance for them without giving inner rank among these requirements. This leads the technique to be suitable for many stakeholders of equal importance [38].

Grey wolf optimization (GWO) is applied by [40] which is one of the most recently meta-heuristic algorithms; in order to prioritize the software project's requirements. In addition; it is evaluated and compared with analytical hierarchy process (AHP) mechanism; where the proposed work performed better than the traditional technique (AHP) by approximately (30%). While, [41] is applied whale optimization algorithm (WOA) which is recently used in optimization problems since it imitates the Humpback whale hunting behavior by employing bubble net hunting technique. It is likewise evaluated with AHP; where the results shown the proposed work outperform the AHP mechanism by approximately (40%).

IV.THE PROPOSED ALGORITHM (WGW)

As shown in Fig.5 the approach that suggested in this study is mainly designed by combining the WOA and the GWO algorithm in order to design a hybrid approach that called "WGW". The combination process is done by combining the advantages of each algorithm that are utilized in this study. The main advantage of GWO that is from each cluster the top three highest values will be considered as α, β and δ and these values represents the first three best solutions from that cluster, with keep into account the group that represents the cluster is limited 5-12 nodes on average. On the other hand the WOA has no limitation in group members and finally the WOA did not have a top three highest solution instead of one. From theses points the WGW is proposed to make a combination between these two algorithms by employing the WOA algorithm to skip the limitation on the group team members and make it unlimited and take the top three highest solutions as GWO works. [12, 13]

As any meta-heuristic optimization algorithms, the proposed algorithm is composed of two stages. The first is the exploration stage while the second is the exploitation stage.

In this study, the proposed WGW algorithm is used to search in the overall search space in order to find all possible X_{rands}, these X_{rands} play an important roles in exploration phase. On the other hand, any search agent sees prey, it is considered to be Xrand, then this foraging process is done by generating bubbles along the path that forms spiral shape and it called the bubble-net hunting technique. Thus, when any search agent sees the bubbles that mean it belongs to this spiral. In case, there are many spirals at the same time from different X_{rand}, each one is considered as cluster in the search space and each one of them has first three best solutions X_α, X_β and X_δ. Whereas this advantage is driven from GWO algorithm, whereas, the previous scenario represents the exploitation phase.[12, 13]

In each iteration, after determining the best solutions X_α, X_β and X_δ, the other search agents update their positions in the cluster. Taken in account the value of absolute A, if it is greater than one means the search agent goes beyond the cluster (spiral) and it must search to another X_{rand} to belong to new cluster. Otherwise, the absolute value of A is less than one means the search agent stills in the cluster and closes to the prey.

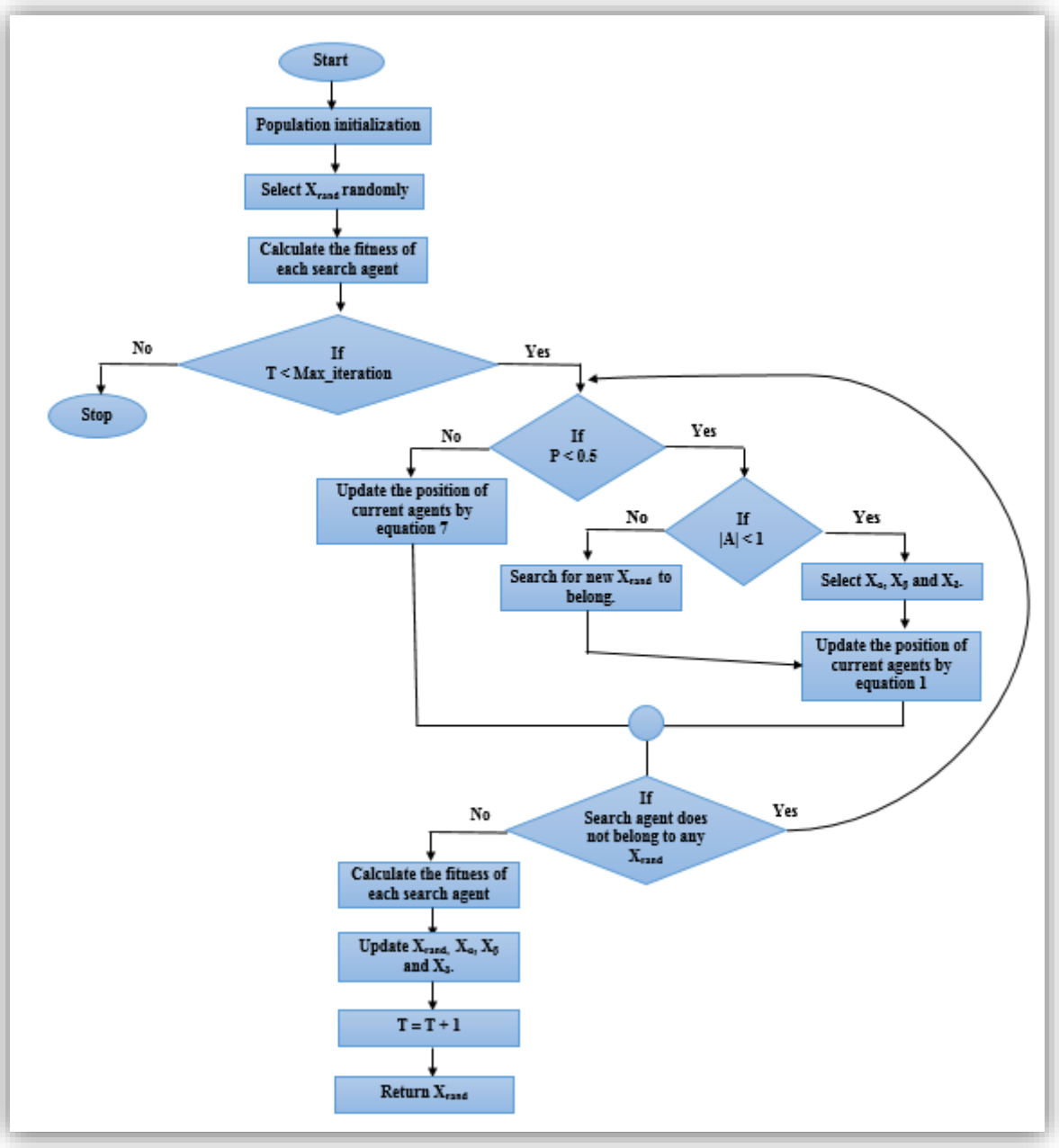

Fig.5. Flowchart of the proposed WGW algorithm.

V.ALGORITHM "RP-WGW"

In this work, an improvement approach is suggested by combining the recently bio-inspired optimization techniques that proposed by Mirjalili et al which are GWO and WO algorithms which imitate the hunting techniques in nature. Then, the WGW algorithm is used for requirements prioritization. To get the prioritization of the requirements, the WGW algorithm is applied. Fig.6 presents the suggested Pseudo Code for "RP-WGW" algorithm.

"RP-WGW" algorithm	
Begin	
1.	Initialize the agents' population Xi (i = 1, 2, 3…..… n)
2.	Initialize C, r and a
3.	Select Xrand randomly
4.	Calculate the distance between each whale (i) and all Xrand by Eq. (1)
5.	If the agent (i) is not assigned
6.	Assign agent (i) to its closet Xrand
7.	Calculate the Fitness for each agent (i)
8.	Invoke Cluster function
9.	Invoke Requirement prioritization function
10.	Return the best solution
End	

Fig. 6. Pseudo – Code for "RP-WGW" algorithm

A.INITIALIZATION STAGE

At the beginning, initialize the agents' population as shown in Fig.6. Number of agents is selected randomly to generate the bubble that performs as cone (spiral) shape. Then, measure the distance between all agents and all X_{rands} to assign to the closet one and join that group to be a member of it.

B.FITNESS FUNCTION

Based on Eq. (8), X_{rand} generates bubble-net when it sight the victim, all agents that see the bubble-net will associate to the cluster (spiral). Thus, the other agents who joined the cluster must update their locations towards the location of X_{rand} as illustrated in Fig. 7. In other hand, these agents will update their locations depending on the value of absolute A. In case, |A| is less than one, this means the agent is still entering the cluster and updates its location by the following Eq. (8). Otherwise, |A| is greater than or equal one, which means the agent is not belong to the cluster, so it search for another X_{rand} to associate by using Eq. (9). this behavior is represented mathematical by Eq. (8, 9) [13] and Eq. (10).

$$\vec{X}\,(t+1) = \vec{D}\,.e^{bl}\,.Cos\,(2\pi l) + \vec{X}\,(t) \tag{8}$$

$$\vec{X}(t+1) = \overrightarrow{Xrand} - \vec{A}.\,\vec{D} \tag{9}$$

$$V = h * r * \pi\,(\frac{1}{3}) \tag{10}$$

Where *r* is the radius of the bubble-net which is constant value (r = 15) [20], *h* is the height of bubble-net, which is selected randomly between 6 and 12 [12, 20], and π is constant value that equal 3.14.

Fitness Function

```
Begin
1.      Xrand creates Bubble-net by Eq. (8)
2.      For each search agent (i)
3.        Update a, A, C and L
4.        Calculate the distance between each agent (i) and Xrand by Eq. (1)
5.        If (A<1)
6.          Update the position of the current agent (i) by the Eq. (8)
7.        Else
8.          Select new Xrand randomly
9.          Update the position of the current agent (i) by the Eq. (9)
10.       End If
11.     End For
End
```

Fig. 7. Fitness Function

C.CLUSTERING

As shown in Fig.8, each cluster K has X_{rand} that selected randomly. The fitness function will be calculated for each agent in order to check if this agent sights the bubble that generated by $X_{rand.}$ In other words, this agent is still joining this spiral. In case the agent didn't see the bubble, it searches for another X_{rand} to belong. Thus, each cluster K will get $X_{\alpha,}$ X_{β} and X_{δ} that represent the first three highest priority of all requirements in this cluster.

Cluster Function

```
Begin
1.      For each cluster K
2.        Select Xrand randomly
3.        Select Xα, Xβ and Xδ
4.        While (t< = max_iteration)
5.          Invoke the fitness function
6.          Invoke the RP function
7.        End While
8.        Return Xα, Xβ and Xδ
9.      End For
End
```

Fig.8. Cluster Function

D.REQUIREMENTS PRIORITIZATION FUNCTION

As discussed in clustering step, each agent in the search space represents a requirement with its importance; this importance represents its importance in the development process of the target project. This importance calculated according to the stakeholders ranking for each requirement with respect the importance of that stakeholder and their effect on the project.

Since each stakeholder belongs to a role in the target environment that the system built for, the role's rank effect directly to the stakeholder importance so when calculating the stakeholder importance the role's rank plays an important role in it.

According to the project environment, first the importance of the role based on the stakeholders ranking on it will be calculated as Equation (11) [39].

$$Influnce\ role(i) = \frac{RRMax + 1 - Rank(Role(i))}{\sum_{J=1}^{n} RRMax + 1 - Rank(Role(J))} \quad (11)$$

Where RRmax is the maximum rank of roles list, Rank (Role (i)) is the rank of the i's role and n is the total number of roles, RRmax+1 is used to invert the value of rank since the lowest rank is highly effect.

After that the influence of each stakeholder in each role will be calculated to find the effect of that stakeholder on the project, Equation (12) [39] shows the stakeholder's importance calculation.

$$Influnce\ (i) = \frac{RSMax + 1 - Rank(i)}{\sum_{K=1}^{n} RSMax + 1 - Rank(K)} \tag{12}$$

Where i represents a specific stakeholder, RSMax is the maximum rank of stakeholder in that role, Rank(i) is the rank of the i'th stakeholder in the role and n is the total number of stakeholders in the same role, RSMax+1 is used to invert the value since the lowest rank is the highest effect.

Then the influence of that stakeholder on the project at all will be calculated by multiplying the role influence and the stakeholder influence as Equation (13) [39].

$$Project\ Influnce\ (i) = \ Influnce\ role(i) \times influnce\ (i) \tag{13}$$

Since each stakeholder ranked the requirement list, the importance of that requirement is calculated by summation of all stakeholders' influence on the project multiplied by its rating on that requirement, Equation (14) shows the requirement importance calculation)[39].

$$Importance\ R = \sum_{i=1}^{n} Project\ influnce(i) \times r(i) \tag{14}$$

Where r (i) represents the i'th stakeholder's rank on that requirement and n is the total number of stakeholders that rating the requirement r (i).

As shown in Fig. 9, the first three top values will be chosen from each cluster. Thus, this process will be executed for each cluster iteratively, as well as a result from this process the set of best solutions from each cluster will be gained in a temporary list, so this list contains the α's, β's and δ's from the clusters. Since this list contains the best solution, the top three solutions will selected as first result of prioritization process. The selected results will be moved to the final ranked list as best three solutions, since they are already taken and prioritized; these requirements will be removed from original clusters.

This process will be repeated until all clusters have no requirements inside, as mention above the selected best solutions will be removed from the original clusters, so the number of requirements inside the cluster will be decreased while iteration keep running.

Finally the proposed approach will return the final ranked set of requirements according to the importance of each requirement.

Requirements Prioritization function

```
Begin
1.      While each cluster is not empty (1,2,3,…K)
2.          For each cluster K
3.              Sort the requirements based on Eq. (14)
4.              Select Xα, Xβ and Xδ and move it to temp_list
5.          End For
6.          Select Xα, Xβ and Xδ  from the sorted temp_list and move them to final
7.  ranked list
8.          Remove Xα, Xβ and Xδ  from original clusters
9.      End While
        Return the Final Results Set
End
```

Fig.9. Requirements Prioritization function

VI.EXPERIMENTAL RESULTS

In order to evaluate the proposed method for requirements prioritization in term of accuracy, the RALIC project was selected from a soo ling lim PhD thesis as a case study in this paper; RALIC stands for the Replacement Access, Library and ID Card project. It was a software project to enhance the existing access control system at University College London (UCL). Located in the central part of London, UCL is concerned with its security [39].

Many UCL buildings require authorized access, such as the libraries, academic departments, and computer clusters. In 2005, UCL had various methods of identification and access control, such as swipe cards, contactless cards, photo ID cards, library barcode, digital security code, and metal door keys. UCL staff and students had to use different mechanisms to access different buildings, which meant they had to carry various cards with them [39]. Furthermore, some of the security systems were already obsolete; others would cease to be operable in a few years' time.

RALIC's aim was to replace the obsolete access control systems, consolidate the various existing access control mechanisms, and at the minimal, combine the photo ID card, access card, and library card [39]. RALIC was a combination of development and customization of an off-the-shelf system [39]. The project started in 2005 and its duration was 2.5 years. The system has been deployed in UCL. The project scope is summarized in Table 1 [39].

Table 1. RALIC Project Scope [39]

Scope Item	Description
1	Replace swipe card readers with smart card readers
2	Source and install access card printers
3	Decide on card design and categories
4	Define user groups and default access rights
5	Provide a more accurate card holder database, save resources on manual data input, and facilitate automated provision and suspension of access and library borrowing rights
6	Issue new cards to staff, students, visitors and contractors
7	Replace the Library access control system
8	Use new cards at the Bloomsbury Fitness Centre

RALIC is a large-scale software project. It had more than 60 stakeholder groups and approximately 30,000 users. Some of the stakeholder groups included students, academic staff, short course and academic visitors, administrators from academic departments, security staff, developers, managers, and front line staff from supporting divisions such as the Estates and Facilities Division that manages UCL's physical estate, Human Resource Division that manages staff information, Registry that manages student information, Information Services Division, and Library Services. RALIC has approximately 30,000 students, staff, and visitors, who use the system to enter UCL buildings, borrow library resources, use the fitness center, and gain IT access.

RALIC had a complex stakeholder base, with different stakeholders having conflicting requirements. For example, members of the UCL Development & Corporate Communications Office preferred the ID card to have UCL branding, but the security guards preferred otherwise for security reasons in case the cards were lost. Some administrators were worried that the new system, which promised to reduce manual labor, would threaten their job. The project involved many divisions in UCL, some of which had low stake in the project but were critical to its success. In particular, the project team found it difficult to engage with the divisions that manage the interfacing systems that supply data to RALIC, such as the Student Registry that provide student data, and Human Resources that provide staff data, because they had little stake in the project. Some representatives from these departments were often absent in project meetings.

The author presents a ground truth in order to evaluate her proposed method; this ground truth contains a real list that prioritized in the real project after deploying it, the data set contains 10 project objectives, 49 requirements and 80 specific requirements after cleaning and processing the data which described in [39].

In this research; the work of [39] was implemented specifically on requirements with keep into account the rank of project objectives and discard her work on specific requirements since the proposed method concerns about requirements prioritization.

The tested data set contains 57 requirements with each importance of them, the result of [39] work is shown in Table 2 which shows a prioritized list of requirements.

Table 2. Prioritized list of requirements [39]

ID	Requirement description	Priority
a.3	all in 1 card	1
a.1	easier to use	2
a.2	use the same access control for library entrance	3
c.3	enable visual checking	4
c.4	access control to include movement tracking/logs	5
c.5	increase access control to buildings	6
c.2	control access to ucl buildings	7
c.1	ensure appropriate access for each individual	8
d.5	granting access rights	9
d.1	faster issue of cards	10
d.2	reduce queuing time	11
d.3	ID card status: Ability to check if a user has collected an ID card	12
d.6	able to create access reports	13
d.4	easy to replace lost cards	14
d.7	procedures for dealing with fraud	15
b.3	card should be secure	16
b.1	card to include user details	17
b.4	card to include ucl branding	18
b.5	easy identification/card is clear looking	19
b.6	card should be sturdy/robust	20
b.2	card to include barcode	21
b.7	card should be attractive	22
g.1	centralized management of access and identification information	23
g.2	export data to other systems	24
g.3	import data from other systems	25
g.4	data access: able to view, update, delete remotely and securely	26
g.5	clear policies on use of access data	27
e.1	save money on cards	28
e.2	save processing time	29
e.3	reduce paper trials	30
f.5	compatible with current network infrastructure	31
f.6	impact to other systems	32
f.4	compatible with UPI	33
f.2	compatible with library systems	34
f.1	compatible with Bloomsbury system (Gladstone MRM)	35
f.3	compatible with HR system	36
h.3	used for computer logon	37
h.2	include payment mechanism	38
h.4	upgradable (software revisions)	39
h.1	include digital certificate	40
h.5	increase security	41
j.2	conform to standards and legislations	42
j.6	Available	43
j.5	Reliable	44

j.3	Technology	45
j.1	fail safe	46
j.7	network infrastructure	47
j.4	Lifecycle	48
j.9	The chosen manufacturer must have a proven tracker record within Institutions with Access Control, ID Pass Production, ODBC, Smart Card technologies.	49
j.10	Photo ID Pass Software must be an embedded feature of the Access control Software and only require software/license upgrades.	50
j.11	The system manufacturer MUST be a Microsoft™ Certified Partner.	51
j.12	The system MUST utilize Microsoft™ Windows 2000 and/or XP Operating System.	52
j.13	The Database platform MUST support Microsoft™ SQL Server and/or Oracle Server.	53
j.8	be capable of having direct printing to both sides of the card, which will include the library barcode	54
i.3	project management activities	55
i.2	technical documents	56
i.1	supplier support	57

In order to evaluate the proposed technique, the theoretical approach was implemented and tested on the same requirements data set with same parameters, the proposed technique prioritize the set of requirements with 52 matching prioritization values and 5 mismatching values. The error rate is calculated using Equation (15) and the accuracy is calculated using Equation (16)

$$\boldsymbol{Error\ Rate\ =\ mismatching/set\ size\ *\ 100} \tag{15}$$

$$\boldsymbol{Accuracy\ =\ 100\%\ -\ Error\ Rate} \tag{16}$$

So the error rate of proposed method is approximately 9% so the approximate accuracy is 91% while comparing the proposed method's result with the used requirement list. Table 3 shows the ordering comparison between the proposed method and [39] work, and Fig.10 shows the error rate and accuracy between the two methods.

Table 3. Comparison of requirements ordering between WGW-RP and work of [39]

WGW ordering (REQ ID)	original ordering (REQ ID)	matching order	WGW ordering (REQ ID)	original ordering (REQ ID)	matching order
a.3	a.3	yes	e.3	e.3	yes
a.1	a.1	yes	f.5	f.5	yes
a.2	a.2	yes	f.6	f.6	yes
c.3	c.3	yes	f.4	f.4	yes
c.4	c.4	yes	f.2	f.2	yes
c.5	c.5	yes	f.1	f.1	yes
c.2	c.2	yes	f.3	f.3	yes
c.1	c.1	yes	h.3	h.3	yes

d.5	d.5	yes	h.2	h.2	yes
d.1	d.1	yes	h.4	h.4	yes
d.2	d.2	yes	h.1	h.1	yes
d.3	d.3	yes	h.5	h.5	yes
d.6	d.6	yes	j.2	j.2	yes
d.4	d.4	yes	j.6	j.6	yes
d.7	d.7	yes	j.5	j.5	yes
b.3	b.3	yes	j.3	j.3	yes
b.1	b.1	yes	j.1	j.1	yes
b.4	b.4	yes	j.7	j.7	yes
b.5	b.5	yes	j.4	j.4	yes
b.6	b.6	yes	j.9	j.9	yes
b.2	b.2	yes	j.11	j.10	no
b.7	b.7	yes	j.8	j.11	no
g.1	g.1	yes	j.10	j.12	no
g.2	g.2	yes	j.12	j.13	no
g.3	g.3	yes	j.13	j.8	no
g.4	g.4	yes	i.3	i.3	yes
g.5	g.5	yes	i.2	i.2	yes
e.1	e.1	yes	i.1	i.1	yes
e.2	e.2	yes			

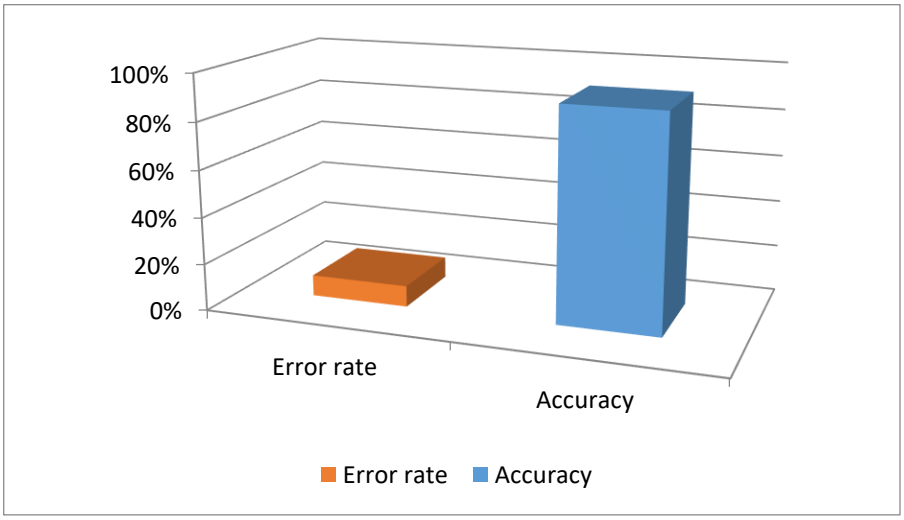

Fig.10. Error rate and Accuracy for WGW-RP

VII. CONCLUSION

Requirement Engineering is the most important phase in software development since it dealing with stakeholders and other activities. Since the number of requirements is varying for each project the requirements prioritization is an important process in order to deliver a good ordering of project's phases with satisfying stakeholders and end users. In this paper a hybrid approach was suggested by combining the advantages of GWO and WOA algorithms as meta-heuristic approach in order to prioritize the software project requirements.

RALIC project's requirements are used to evaluate the proposed method (WGW). The WGW shows 91% accuracy and 9% Error rate of prioritizing these requirements compared to work of author [39].

REFERENCES

[1] Hasan, M. S., Mahmood, A. A., Alam, M. J., Hasan, S. N., & Rahman, F. (2010). An evaluation of software requirement prioritization techniques. *International Journal of Computer Science and Information Security (IJCSIS)*, 8(9).

[2] Duan, C., Laurent, P., Cleland-Huang, J., & Kwiatkowski, C. (2009). Towards automated requirements prioritization and triage. Requirements engineering, 14(2), 73-89.

[3] Paetsch, F., Eberlein, A., & Maurer, F. (2003, June). Requirements engineering and agile software development. In *Proceedings of the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03)* (pp. 308-313). IEEE.

[4] Wiegers, K. (1999). First things first: prioritizing requirements. *Software Development*, 7(9), 48-53.

[5] Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3), 264-323.

[6] Emami, H., & Derakhshan, F. (2015). Integrating fuzzy K-means, particle swarm optimization, and imperialist competitive algorithm for data clustering. *Arabian Journal for Science and Engineering*, 40(12), 3545-3554.

[7] Yang, F., Sun, T., & Zhang, C. (2009). An efficient hybrid data clustering method based on K-harmonic means and Particle Swarm Optimization. *Expert Systems with Applications*, 36(6), 9847-9852.

[8] Nayak, J., Naik, B., & Behera, H. S. (2015). Fuzzy C-means (FCM) clustering algorithm: a decade review from 2000 to 2014. In *Computational intelligence in data mining*-volume 2 (pp. 133-149). Springer, New Delhi.

[9] Kumar, Y., & Sahoo, G. (2015). Hybridization of magnetic charge system search and particle swarm optimization for efficient data clustering using neighborhood search strategy. *Soft Computing*, 19(12), 3621-3645.

[10] Alpaydin, E. (2004*). Introduction to machine learning*. Cambridge: MIT press.

[11] Zhang, Q. H., Li, B. L., Liu, Y. J., Gao, L., Liu, L. J., & Shi, X. L. (2016). Data clustering using multivariant optimization algorithm. *International Journal of Machine Learning and Cybernetics*, 7(5), 773-782.

[12] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software*, 69, 46-61.

[13] Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, 95, 51-67.

[14] Chander, S., Vijaya, P., & Dhyani, P. (2018). Multi kernel and dynamic fractional lion optimization algorithm for data clustering. *Alexandria engineering journal*, 57(1), 267-276.

[15] Çomak, E. (2016). A modified particle swarm optimization algorithm using Renyi entropy-based clustering. *Neural Computing and Applications*, 27(5), 1381-1390.

[16] Greer, D., & Bustard, D. W. (1997). SERUM-Software engineering risk: Understanding and management. *The International Journal of Project & Business Risk*, 1(4), 373-388.

[17] Ma, Q. (2009). The effectiveness of requirements prioritization techniques for a medium to large number of requirements: a systematic literature review (*Doctoral dissertation*, Auckland University of Technology).

[18] Ibrahim, O., & Nosseir, A. (2016). A Combined AHP and Source of Power Schemes for Prioritising Requirements Applied on a Human Resources. In *MATEC Web of Conferences* (Vol. 76, p. 04016). EDP Sciences

[19] Goldbogen, J. A., Friedlaender, A. S., Calambokidis, J., Mckenna, M. F., Simon, M., et al. (2013). Integrative approaches to the study of baleen whale diving behavior, feeding performance, and foraging ecology. *BioScience*, 63(2), 90-100..

[20] Humpback whale. (n.d.). In Wikipedia. Retrieved August 20, 2018, from https://en.wikipedia.org/wiki/Humpback_whale

[21] Daneva, M., Damian, D., Marchetto, A., & Pastor, O. (2014). Empirical research methodologies and studies in Requirements Engineering: How far did we come?. *Journal of systems and software*, 95, 1-9.

[22] Achimugu, P., Selamat, A., Ibrahim, R., & Mahrin, M. N. R. (2014). A systematic literature review of software requirements prioritization research. *Information and software technology*, 56(6), 568-585.

[23] Karlsson, J., Wohlin, C., & Regnell, B. (1998). An evaluation of methods for prioritizing software requirements. *Information and software technology*, 39(14-15), 939-947.

[24] Brender, S. Key words for use in RFC's to indicate requirements levels. RFC 2119.

[25] D. Leffingwell & D. Widrig. (1999) *Managing Software Requirements: A Unified Approach*, Upper Saddle River:Addison- Wesley.

[26] Hatton,S. (2007). Early prioritisation of goals. In *International Conference on Conceptual Modeling* (pp. 235-244). Springer, Berlin, Heidelberg.

[27] IEEE Std 830-1998 (1998) IEEE recommended practice for software requirements specifications. IEEE Computer Society, Los Alamitos

[28] Sommerville, I., & Sawyer, P. (1997). *Requirements engineering: a good practice guide*. John Wiley & Sons, Inc..

[29] Regnell B, Höst M, Natt och Dag J, Beremark P, Hjelm T (2001) An industrial case study on distributed prioritization in market-driven requirements engineering for packaged software. *Requirements Engineering* 6(1):51-62

[30] Saaty TL (1980*) The analytic hierarchy process*. McGraw-Hill, New York

[31] Lehtola, L., & Kauppinen, M. (2004). Empirical evaluation of two requirements prioritization methods in product development projects. *In European Conference on Software Process Improvement* (pp. 161-170). Springer, Berlin, Heidelberg.

[32] Karlsson, J., Wohlin, C., & Regnell, B. (1998). An evaluation of methods for prioritizing software requirements. *Information and software technology*, 39(14-15), 939-947.

[33] Shen, Y., Hoerl, A. E., & Mcconnell, W. (1992). An incomplete design in the analytic hierarchy process. *Mathematical and Computer Modelling: An International Journal*, 16(5), 121-129.

[34] Harker, P. T. (1987). Incomplete pairwise comparisons in the analytic hierarchy process. *Mathematical Modelling*, 9(11), 837-848.

[35] Karlsson, J., Olsson, S., & Ryan, K. (1997). Improved practical support for large-scale requirements prioritising. *Requirements Engineering*, 2(1), 51-60.

[36] Dsdm Public version 4.2, from www.dsdm.org, Tech. Rep., Retrieved, 6 June, 2009, last visited, April, 5, 2018.

[37] Aho, A. V., Hopcroft, J. E., & Ullman, J. (1983). *Data structures and algorithms*. Addison-Wesley Longman Publishing Co., Inc..

[38] Lauesen, S. (2002) *Software requirements – styles and techniques*. Pearson Education, Essex

[39] Lim, S. L. (2011). *Social networks and collaborative filtering for large-scale requirements elicitation* (Doctoral dissertation, University of New South Wales).

[40] Alzaqebah, A., Masadeh, R., & Hudaib, A. (2018). Whale Optimization Algorithm for Requirements Prioritization. In *Proceedings of the 9th International Conference on Information and Communication Systems (ICICS)*, (pp. 84-89). IEEE.

[41] Masadeh, R., Alzaqebah, A. & Hudaib, A. (2018). Grey Wolf Algorithm for Requirements Prioritization. *Modern Applied Science*, 12(2), 54.

[42] Hudaib, A., Masadeh, R., Qasem, M. H., & Alzaqebah, A. (2018). Requirements Prioritization Techniques Comparison. *Modern Applied Science*, 12(2), 62.

[43] Masadeh, R., Alzaqebah, A., & Sharieh, A. (2018). Whale Optimization Algorithm For Solving The Maximum Flow Problem. *Journal of Theoretical & Applied Information Technology*, 96(8).

[44] Masadeh, R., Sharieh, A., & Sliet, A. (2017). Grey wolf optimization applied to the maximum flow problem. *International Journal of Advanced and Applied Sciences*, 4(7), 95-100.

[45] Yassien, E., Masadeh, R., Alzaqebah, A., & Shaheen, A. (2017). Grey Wolf Optimization Applied to the 0/1 Knapsack Problem. *International Journal of Computer Applications*, 169(5).

[46] Tarhini, A., Ammar, H., & Tarhini, T. (2015). Analysis of the critical success factors for enterprise resource planning implementation from stakeholders' perspective: A systematic review. *International Business Research*, 8(4), 25.

[47] Vestola, M. (2010). A comparison of nine basic techniques for requirements prioritization. Helsinki University of Technology.

