International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 14, No.1 (Jun-24)

http://dx.doi.org/10.12785/ijcds/1501122

Reptile Search Algorithm for Association Rule Mining

Abderrahim Boukhalat[1,2], KamelEddine Heraguemi[3], Mohamed Benouis[1], Brahim Bouderah[1] and Samir Akhrouf[1]

[1]*Computer Science,University Mohamed Boudiaf, M'Sila, Algeria*
[2]*LIM Laboratory,University of Souk Ahras, Algeria*
[3]*National School of Artificial Intelligence, Algiers, Algeria*

Received 2 Nov. 2023, Revised 2 Apr. 2024, Accepted 6 Apr. 2024, Published 1 Jun. 2024

Abstract: Association rule mining (ARM) is a very popular, engaging, and active research area in data mining. It seeks to find valuable connections between different attributes in a defined dataset. ARM, which describes it as an NP-complete problem, creates a fertile field for optimization applications. The Reptile Search Algorithm (RSA) is an innovative evolutionary algorithm. It yanks stimulation from the encircling and hunting conducts of crocodiles. It is a well-known optimization technique for solving NP-complete issues. Since its introduction by Abualigah et al. in 2022, the approach has attracted considerable attention from researchers and has extensively been used to address diverse optimization issues in several disciplines. This is due to its satisfactory execution speed, efficient convergence rate, and superior effectiveness compared to other widely recognized optimization methods. This paper suggests a new version of the reptile search algorithm for resolving the association rules mining challenge. Our proposal inherits the trade-off between local and global search optimization issues demonstrated by the Reptile search algorithm. To illustrate the power of our proposal, a sequence of experiments is taken out on a varied, well-known, employing multiple comparison criteria. The results show an evident dominance of the proposed approach in the front of the famous association rules mining algorithms as well as Bees Swarm Optimization (BSO), Bat Algorithm (BA), Whale Optimization Algorithm (WOA), and others regarding CPU time, fitness criteria, and the quality of generated rules.

Keywords: Data mining, Association rule mining, Bio-Inspired Approaches, Reptile Search Algorithm.

1. Introduction

Over the past few decades, the exponential expansion of stored data has sparked growing interest among researchers in extracting valuable information from data. Knowledge discovery in databases (KDD) [1] is a comprehensive procedure for removing helpful information from raw data. This holistic approach transforms data into understandable and actionable information, a crucial step in converting data into valuable insights. KDD includes data collection, preprocessing, data mining, and result interpretation. However, Data Mining is still the primary step in data knowledge discovery, and it defines algorithms and techniques for processing practice of revealing concealed patterns, trends, and valuable insights within extensive datasets containing structured or unstructured data. Its applications span diverse fields such as business intelligence, marketing, healthcare, and finance, facilitating well-informed decision-making and predictive analytics.

One of the most crucial processes in data mining is Association rule mining (ARM) [2]. It seeks to extract proper relationships between elements in stored data. Association rules are ideal and valuable for various applications, such as business and marketing, healthcare diagnosis, and telecommunications. These rules have equipped managers with useful insights to mold their marketing strategies and boost their overall gains. Researchers began to show significant interest in Association Rule Mining (ARM) after its introduction by Agrawal et al. [2] at the beginning of the 90s. ARM's primary objective is to discover connections between items in real-world databases. Initially created for market basket analysis, its goal was to reveal links between products, like the well-known case of "milk $\Rightarrow$ bread," indicating that buying milk often coincides with purchasing bread. Researchers create numerous traditional algorithms, such as Apriori [3] and FP-growth [4], to solve ARM problems. In the mathematical presentation of ARM, the process begins with analyzing transactional data, where each transaction comprises a collection of items. ARM aims to identify frequent item sets, subsets of items that appear together in many transactions. Typically, researchers measure this frequency using support and confidence metrics. These approaches extract all the possible relations between the attributes in the data sets. Add on one side or delete it. Algorithms require massive processing resources and often

E-mail address: abderrahim.boukhalat@univ-msila.dz, kameleddine.heraguemi@ensia.edu.dz, mohamed.benouis@univ-msila.dz,brahim.bouderah@univ-msila.dz,samir.akhrouf@univ-msila.dz

fail as the data size increases [5], which takes a lot of time and grows memory space [6].

Researchers have turned to applying meta-heuristic approaches to address the limitations of exact algorithms in association rule mining (ARM). These techniques, previously employed to tackle many NP-complete problems, are relevant to ARM because it is also an NP-complete problem. Numerous studies have explored the potential of evolutionary and swarm-inspired algorithms to identify optimal rules in ARM. One of the earlier approaches was using genetic algorithms [7], which has shown promise in optimizing association rule mining. In addition to genetic algorithms, swarm intelligence techniques have gained traction in ARM. Researchers have experimented with various well-established swarm-inspired algorithms, such as the Binary Particle Swarm Optimization Algorithm(BPSO) [8], Bees Swarm Optimization (BSO) [9], Bat Algorithm (BA) [10], and Whale Optimization Algorithm (WOA) [11]. These methods draw inspiration from the collective behavior of natural systems, such as bird flocks, bee colonies, and whales' humpback hunting behavior, to efficiently explore and refine rule sets. In ARM, datasets serve as the sample search space, and the primary objective of intelligent algorithms is to optimize an underlying mathematical function by maximizing or minimizing it. This function evaluates the quality of solutions based on various measurements. By employing meta-heuristic techniques, researchers aim to navigate the vast search space of possible association rules more effectively, improving the accuracy and applicability of the identified rules in large and complex datasets. These bio-inspired approaches represent a significant advancement in association rule mining, offering perspectives and solutions to overcome the inherent challenges of this NP-complete problem. Laith Abualigah et al. introduced a natural process or heuristic approach known as RSA (Reptile Search Algorithm) in 2022. [12].

The Reptile Search Algorithm attempts to address complicated optimization issues. It aims to imitate crocodiles' hunting behavior to find optimal solutions. The Algorithm incorporates two critical aspects of crocodile behavior into its operation: encircling and hunting. Encircling is conducted through either belly walking or high walking, whereas hunting is executed using either hunting coordination or hunting cooperation strategies. This Algorithm aims to discover optimal solutions for a diverse set of optimization issues by simulating the hunting behaviors observed in crocodiles. The RSA algorithm has demonstrated its effectiveness in addressing various optimization problems. The Algorithm exhibits versatility as it can address multiple issues, including unconstrained and constrained problems, single-objective and multi-objective problems, and difficulties involving continuous and discrete variables. Many fields, including finance, image, and signal processing [13], engineering [14], renewable energy [15], machine learning, and many others, have adopted it due to its exceptional efficiency and robustness. This is because it outperforms other popular optimization methods, has a sufficiently fast execution time, and has a good quality convergence rate.

Furthermore, this technique exhibits superiority in simplicity compared to other meta-heuristics. In this paper, we propose RSA-ARM, a novel approach that applies the principles of the RSA algorithm to association rule mining. Our research aims to fill a critical gap in the literature by investigating the effectiveness of RSA in extracting association rules from diverse datasets. Our methodology introduces several innovations, including a low-consuming solution generation method based on association rule definitions and vertical dataset presentation to improve computational efficiency.

As a summary, the key contributions in this paper are:

- We investigate the application of the RSA algorithm, a novel approach designed explicitly for association rule mining. This research fills a critical gap in the literature, as prior studies haven't explored the effectiveness of RSA for mining association rules.
- The developed RSA-ARM is experimented on benchmark datasets of various sizes, small, medium, and large, to assess its effectiveness over existing association rule mining techniques regarding memory use, CPU runtime, number of generated rules, and quality.

We organize the remains of this paper as follows: the next part summarizes the latest advancements in association rule mining and evolutionary algorithms. Section 3 provides a broad perspective of the association rule mining issue and its basics, along with an overview of RSA's original approach. Section 4 details the principles and strategies of RSA-ARM. Section 5 provides the experimental findings, encompassing a range of comparative analyses with recently developed cutting-edge algorithms. Finally, the last section delves into the drawn conclusion and outlines future possibilities and directions.

2. Background

A. Association Rule Mining

Agrawal first proposed ARM, also called the market basket issue, as a fundamental research task in data mining. Researchers use Association Rule Mining (ARM) to discover important relationships and patterns between items in a dataset, particularly in the context of sales transactions. ARM aims to discover valuable associations or rules that reveal meaningful connections between elements or features within the dataset [1]. A database (D) comprises n unique items and m transactions. The set of items is denoted as $I = \{i_1, i_2, \ldots, i_n\}$, while the database is composed of transactions $D = \{t_1, t_2, \ldots, t_m\}$. Each transaction t represents a subset of items from the set I, indicated as ($t \subseteq I$). Association rules, exemplified by $X \rightarrow Y$, are generated from the transaction database, relying on support and confidence metrics. X and Y represent subsets of I, and there are no shared items between them ($X \cap Y = \emptyset$). X is specifically identified as the antecedent, while Y is denoted as the consequent of the rule.

Definition 1. The support of an association rule $X \rightarrow Y$ is determined by dividing the occurrence frequency of the combined itemset $X \cup Y$ by the entire number of transactions $|D|$ in the database. Eq. (1) describes this calculation.

$$\text{Support}(X \rightarrow Y) = \frac{P(X \cup Y)}{|D|} \tag{1}$$

Definition 2. The confidence (Conf) for the rule $X \rightarrow Y$, as outlined in Eq. (2), quantifies the proportion of transactions within dataset D that simultaneously include itemset X and itemset Y. This measure quantifies the confidence level in the association between item sets X and Y.

$$\text{Confidence}(X \rightarrow Y) = \frac{P(X \cup Y)}{P(X)} \tag{2}$$

B. *The Reptile Search Algorithm (RSA)*

Abualigah et al. [12] presented the RSA approach in 2022, which applies to exploration and exploitation. The RSA Algorithm is influenced by the natural behaviors of crocodiles, which encompass encircling, hunting, and social interactions. These innate mechanisms translate into mathematical models to create the RSA algorithm, which is subsequently employed in optimization procedures. Figures 1 and 2 visually represent the Algorithm's two mechanisms of encircling and hunting. As described in the paper by Abualigah et al. [12], the Algorithm consists of three distinct phases:

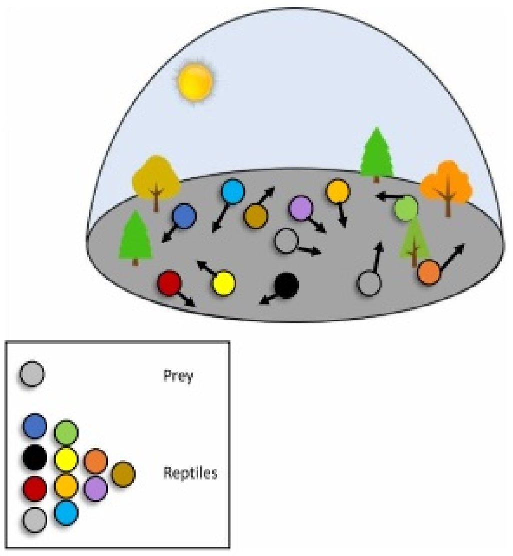


Figure 1. Encircling the prey (Exploration) [12]

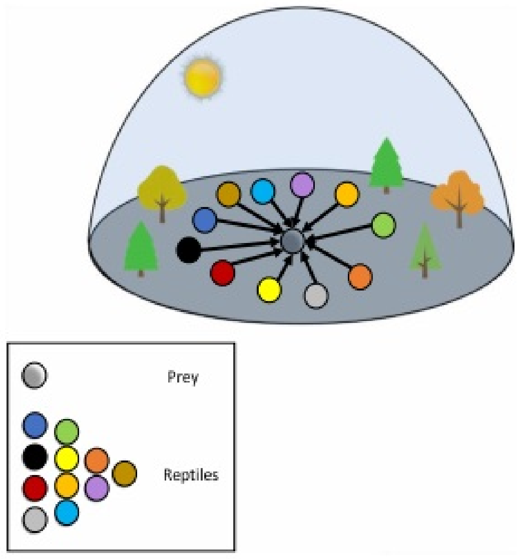


Figure 2. Hunting phase for attacking the prey (Exploitation) [12]

1) *Initialization Phase*

As demonstrated in Eq. (3), RSA's optimization procedure commences with an initial set of randomly generated candidate solutions denoted as X, and the best-obtained solution remains highly close to optimal in each iteration.

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & x_{1,n-1} & x_{1,n} \\ x_{2,1} & \cdots & x_{2,j} & \cdots & x_{2,n} \\ \cdots & \cdots & x_{i,j} & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N-1,1} & \cdots & x_{N-1,j} & \cdots & x_{N-1,n} \\ x_{N,1} & \cdots & x_{N,j} & x_{N,n-1} & x_{N,n} \end{bmatrix} \tag{3}$$

Using Eq. (4), X represents a collection of randomly generated candidate solutions, Xi,j describes the position of solution i in the j proportions, N signifies the count of possible resolutions, and n represents the size of the issue dimension.

$$x_{ij} = \text{L}\,B + rand \times (UB - LB), j = 1, 2, \ldots, n \tag{4}$$

In which rand represents a randomly generated number, while UB and LB refer to the upper bounds and lower bounds of the specific issue under consideration, respectively.

$$x_{i,j}(1+v) = \begin{cases} [\beta \times -\eta_{i,j}(v) \times Best_j(v)] - \text{ rand } \times R_{i,j}(v), \\ \quad t \le \frac{T}{4} \\ \text{E}S\,(v) \times Best_j(v) \times \text{ rand } \times x_{r_1,j}, \\ \quad v \le 2\frac{T}{4} \text{ and } v > \frac{T}{4} \end{cases} \tag{5}$$

2) *Exploration Phase (Encircling)*

The RSA algorithm employs a flexible switching mechanism between its exploration and exploitation search phases, guided by four specific conditions. This shift is made smoother by dividing the overall iteration count by four. Belly and high walking are the primary search methods used during the RSA search stages. These strategies are instrumental in exploring search regions and uncovering superior solutions, with each search phase governed by two distinct conditions. The strategy of high walking is conditioned by $v \leq \frac{T}{4}$ while belly walking is conditioned by $v \leq 2\frac{T}{4}$ and $v > \frac{T}{4}$. The equations specifying the update of positions during the exploration phase are in Eq. (5). In this formula, $Best j(v)$ represents the position achieved at the jth iteration v. $rand$ is a randomly generated number between zero and 1.v, representing the present iteration number, while T is the highest number of iterations. These equations determine the update of positions during the Algorithm's exploration phase of the Algorithm. $\eta_{i,j}$ is the operator of hunting, which is calculated using Eq. (6). β is equal to 0.1. An arbitrary coordinate is denoted by $X_(r1, j)$, where $r1$ is an arbitrary integer in the range [1, N]. N is the number of the nominee solutions. The value of the reduce function, denoted as $R_{(i,j)}$, is computed using Eq. (7). Evolutionary Sense ($ES(v)$) be calculated using Eq. (8).

$$\eta_{i,j} = \mathrm{P}_{i,j} \times Best_j(v) \tag{6}$$

$$R_{i,j} = \frac{\mathrm{Best}_j(v) - x_{r_2,j}}{\mathrm{Best}\ _j(v) + \epsilon} \tag{7}$$

$$ES(v) = 2 \times \left(1 - \frac{1}{T}\right) \times r_3 \tag{8}$$

Additionally, ϵ represents a little value, $r2$ is an arbitrary number selected from the interval $[1, N]$, and $r3$ denotes a haphazardly developed integer within the range of -1 to 1. Pij indicates the rate variation between the jth position of the best solution got and the jth position of the present solution, with this percentage difference computed according to Eq. (9).

$$P_{i,j} = \alpha + \frac{x_{i,j} - M(x_i)}{\left(UB_{(j)} - LB_{(j)}\right) \times \ \mathrm{Best}\ _j(v) + \epsilon}, \tag{9}$$

Here, $M(xi)$ represents the average positions calculated using Eq. (10). $UB(j)$ and $LB(j)$ represent the upper and lower bounds, respectively, with α set to a set value of 0.1.

$$M(x_i) = \frac{1}{n} \sum_{j=1}^{n} x_{i,j}, \tag{10}$$

3) Exploitation Phase (Hunting)

The RSA's exploitation mechanisms employ two primary search strategies, hunting coordination and cooperation, to locate the optimal solution, as shown in Eq (11). In this phase, the hunting coordination approach depends on conditions $v \leq 3\frac{T}{4}$ and $v > 2\frac{T}{4}$. Otherwise, the hunting cooperation technique is utilized if $v \leq T$ and $v > 3\frac{T}{4}$. In light of the preceding discussion, Algorithm 1 details the RSA algorithm.

$$x_{i,j}(1+v) = \begin{cases} P_{i,j}(v) \times \ \mathrm{rand}\ \times \ \mathrm{Best}\ _j(v), \\ \quad v \leq 3\frac{T}{4} \text{ and } v > 2\frac{T}{4} \\ \mathrm{Best}\ _j(v) - \eta_{i,j}(v) \times \epsilon - \ \mathrm{rand}\ \times R_{i,j}(v), \\ \quad v \leq T \text{ and } v > 3\frac{T}{4} \end{cases} \tag{11}$$

Algorithm 1 Reptile Search Algorithm (RSA)

```
Require: Max iterations $T$ and the population, etc.
 1: Initialize RSA criterion $\alpha$, $\beta$, etc.
 2: Initialize randomly the solutions $X$: $i = 1, \dots, N$.
 3: while $v < T$ do
 4:     Compute the Fitness for the nominee solutions ($X$).
 5:     Determine the current best solution.
 6:     Modify the $ES$ using Equations (8).
 7:     The start of the RSA.
 8:     for $i = 1$ to $N$ do
 9:         for $j = 1$ to $n$ do
10:             Update the $R$,$P$, and $eta$ using Equations (6), (7), and (9), respectively.
11:             if ($v \leq \frac{T}{4}$) then
12:                 $x_{i,j}(1+v) = [\beta \times -\eta_{i,j}(v) \times Best_j(v)] -$ rand $\times R_{i,j}(v)$   // High walking
13:             else if ($v \leq \frac{2T}{4}$ and $v > \frac{T}{4}$) then
14:                 $x_{i,j}(1+v) = ES(v) \times Best_j(v) \times$ rand $\times x_{r_1,j}$   // Belly walking
15:             else if ($v \leq \frac{3T}{4}$ and $v > \frac{2T}{4}$) then
16:                 $x_{i,j}(1+v) = P_{i,j}(v) \times$ rand $\times$ Best $_j(v)$   // Hunting coordination
17:             else
18:                 $x_{i,j}(1+v) =$ Best $_j(v) - \eta_{i,j}(v) \times \epsilon -$ rand $\times R_{i,j}(v)$   // Hunting cooperation
19:             end if
20:         end for
21:     end for
22:     $v = v + 1$
23: end while
24: Return the best solution (Best($X$)).
```

3. Related Work

The main goal of association rule mining (ARM) is to discover rules that fulfill predetermined criteria, typically the lowest support and confidence thresholds, in a provided dataset. The two most widely used ARM algorithms are the FP-growth and Apriori algorithms. These methods work by analyzing all datasets, detecting frequent item sets that exceed the lowest support requirement, and deriving ARM established on the lowest confidence threshold [16]. In the subsequent stages of research, several novel algorithms have emerged as alternatives to traditional Apriori and FP-growth algorithms, addressing some of the inherent challenges in those approaches. Notable algorithms in this category include Eclat [17], Charm [18], and Modified FP-growth (MFP-growth) algorithm [19], designed to improve the FP-growth effectiveness by eliminating the requirement for the recurrent result of dependent subtrees. Employing a header table configuration, the proposed Algorithm reduces the overall complexity of the frequent pattern tree. Wu et al.[20] proposed enhancing the Apriori algorithm to address its inefficiency in mining association rules from large databases. They introduced a genetic algorithm-based improvement, outlining the basic principles of association rules and the Apriori algorithm. The proposed enhancement, based on the Partition algorithm, is detailed along with modifications to the genetic Algorithm's coding scheme and fitness function.

In [21], the paper presented an optimized ARM algorithm to address the Apriori algorithm's limitations. It improves the efficiency and accuracy of recommendation systems in e-commerce. By combining a multi-item support tree with dynamic support adjustment, the Algorithm enhances the generation of frequent item sets and adapts the minimum support threshold during mining. Authors in [22] introduced prior information by incorporating standard association rules into the database through tagging. It reduces the known rules' combinations during algorithm operation to decrease space complexity and improve runtime efficiency. The study primarily examines association rules within shopping baskets using experimentally generated orders. In 2023, Zhang et al. [23] proposed an algorithm that utilizes a double support degree and compression matrix approach to enhance drug search efficiency. The Algorithm reduces matrix scans and candidate set generation by incorporating the compression matrix principle and optimizing matrix storage using arrays. Moreover, it applies support threshold constraints to limit infrequent and overfrequent itemset generation, ensuring rule validity and practicality.

Despite these significant advancements, the current generation of algorithms continues to grapple with challenges when handling the enormous volumes of data generated in today's digital era. This often results in slower processing times and significant memory consumption issues. However, traditional ARM algorithms usually exhibit efficiency and memory consumption limitations when dealing with extensive datasets. Researchers have explored integrating optimization techniques with ARM to address these challenges. Previous research has shown that meta-heuristic methods based on populations, such as genetic or swarm-inspired algorithms, are good ways to solve problems with association rule mining [24].In essence, the marriage between optimization techniques and ARM aims to enhance the performance and scalability of rule discovery in large datasets. This strategy efficiently utilizes meta-heuristic algorithms to navigate the complex search space of possible association rules. This leads to rule-based mining processes that are more effective and reliable. Wang and Bridges were the first to describe formally the use of GA for ARM. [25].

In pioneering research, Haldulakar et al. achieved a significant milestone by enhancing the utility of an evolutionary Algorithmin ARM. This groundbreaking work has paved the way for numerous subsequent studies and innovations in the field. Specifically, they applied a genetic approach to enhance the results obtained from the Apriori algorithm. The primary goal was to generate association rules that are not only reliable but also highly valuable for the end user [26]. Association rules with GA (ARMGA) [26] is a widely recognized method used for mining Binary Association Rules (BARs). However, a significant drawback of this approach is its tendency to produce solutions that do not meet admissibility criteria. There have been efforts to enhance ARMGA to address this limitation. In [27], researchers utilized a genetic algorithm to extract association rules and identify a strong correlation between critical elements. Researchers use numerous bio-inspired techniques, including genetic algorithms, to discover association rules. In [28], they construct the G3PARM technique through genetic programming (GP). The authors utilized grammar-guided genetic programming (G3P) to filter out potentially invalid individuals during the operation of genetic programming (GP).

Researchers applied ARM methods based on swarm intelligence, including the bat Algorithm (BA), the Ant Colony Optimization (ACO) [29], Bee Swarm Optimization (BSO), Cuckoo Search (CS) [30], Particle Swarm Optimization (PSO) [31], and the Penguin Search Optimization Algorithm (Pe-SOA) [32], to ARM tasks. These methods have demonstrated improved runtime performance, but there is still room for enhancing the quality of the solutions they generate. In [31], researchers employ PSO as an innovative technique in ARM. The PSO-based approach consists of two main stages: the preprocessing step and the mining step. These stages serve distinct purposes in the process of finding association rules. They have developed an enhanced version of this work based on PSO [33]. It introduces a binary variant of PSO called BPSO, which they use to mine association rules. It generates the best rule X without performing all measurement levels, where X is the threshold's performance. Penguins exhibit hunting behaviors that inspire the Pe-SOA algorithm. Gheraibia et al. use Pe-SOA for ARM to expedite the exploration process [32]. They integrated an overlapping distance measure to produce non-repetitive rules with minimal overlap with the existing rules.

In the same line, Djenouri et al. [34] introduced the BSO-ARM, which employed a removal and simplification strategy. Djenouri et al. [35], [8], presented two BSO-ARM adjustments. They used three heuristics to investigate space search in the initial improvement. The second improvement leverages the recursive nature of frequent items (FIs), focusing on the examination and study of bees' regions instead of conducting a localized search. Heraguemi et al. presented Bat-ARM, an innovative version of BA for ARM [36]. They improved BAT-ARM using a master/slave technique by splitting the population into smaller groups [37]. The lack of inter-bat communication in the Bat-ARM Algorithm limits the search space that the bats can explore. In considering this problem, Heraguemi et al. [38] proposed three solutions: the ring, the master-slave, and the hybrid.

Since ARM is a four-objective problem, Heraguemi et al. [39] looked at the feasibility of using multi-objective BA to solve it. A rule miner for binary cuckoo searches is introduced [30]. To address the ARM problem, the authors of this approach adjusted the cuckoo search method into a binary variant by incorporating the sigmoid operation.

In a more recent development, Heraguemi et al. [11] introduced an innovative approach named Whale Optimization Algorithm for Association Rule Mining "WO-ARM". It represents a unique fusion of the WOA with association rule mining techniques, aiming to harness the strengths of both methods for better results. This approach leverages the characteristics of the Whale Optimization Algorithm to enhance discovering association rules. Its goal is to improve efficiency and rule quality extracted from datasets. Nonetheless, WOA-ARM can often get in local optima. Consequently, researchers introduced an enhanced version called the WOA-based adaptive criterion method and Levy flight approach (LWOA) [40]. Researchers presented a novel hybrid Association Rule Mining (ARM) approach, which combines water wave optimization (WWO) with Levy flight, in [41]. This method enhances the capabilities of WWO by accelerating the search process and effectively broadening the exploration of the search space through Levy flight. The study employs a hybrid approach that combines Bat Algorithm (BA), Ant Colony Optimization (ACO), and Cuckoo Search (CS) with Levy flight Water Wave Optimization (LWWO), referred to as LWWO-ACO, LWWO-BA, and LWWO-CS, respectively. Alongside these hybrid algorithms, other methodologies are also considered. This approach significantly enhances population diversity, ultimately enabling the discovery of optimal solutions on a global scale.

Moulai et al.[42] introduced two innovative swarm intelligence approaches tailored for discrete problems. Inspired by the behavior of elephants, two continuous evolutionary algorithms, namely elephant herding optimization and elephant swarm water search algorithm, are adapted into discrete versions: discrete elephant herding optimization and discrete elephant swarm water search algorithm, which are employed to extract association rules from large-scale databases. Wang et al.[43] introduced an intelligent evolutionary algorithm. The study transforms the Market Basket Analysis problem into a multi-objective optimization task to recognize the drawbacks of single-objective evolutionary algorithms. It proposes a multi-level association rule mining algorithm based on NSGA-II. The approach combines the Apriori algorithm for frequent item set generation with NSGA-II for association rule mining, enhancing computational efficiency. Babak Rokh et al.[44] introduce a novel MOFNAR approach for efficiently mining association rules in numerical data that employs a multi-objective firefly algorithm to balance simplicity and accuracy in rule extraction. By incorporating objectives such as Balance, Square of Cosine (SOC), and comprehensibility, MOFNAR aims to generate rule sets that are both simple and accurate. The method's effectiveness may depend heavily on the choice of parameters and objectives, which could affect the quality of the generated rule sets. Additionally, the scalability of MOFNAR to massive datasets or datasets with high dimensionality may be limited, potentially leading to longer execution times or computational efficiencies. Carlos et al.[45] focused on addressing the challenges posed by the increasing volume of data in association rule mining. The study reviewed existing algorithms for frequent item set and association rule mining and proposed new efficient algorithms for handling large datasets. Utilizing distributed computation, the study develops novel algorithms for frequent itemset extraction and association rule mining in Spark.

Abdelkader Mokkadem et al.[46] present PrefRec, an algorithm for discovering frequent item sets and association rules. Its primary strength lies in its recursive nature concerning the items, making it highly efficient for updating the mining process when new items are introduced or existing ones are removed from the database. Also, they[47] introduce SufRec, a novel algorithm for mining frequent item sets and association rules. SufRec offers two versions: SufRecDep and SufRecInd. The former proceeds through tasks sequentially, leveraging results from previous tasks, while the latter executes tasks independently. Both versions exhibit a recursive nature concerning items, allowing for efficient updates to the mining process when new items are introduced or excluded from the database. In [48], Siswanto et al. proposed a novel SDFP-growth algorithm that automates determining the support threshold and doesn't necessitate manual selection of the support threshold value. By performing dimensionality reduction on the original dataset, the Algorithm generates smaller datasets with optimized support thresholds, thereby improving efficiency and reducing computational burden. Researchers proposed a novel approach for association rule mining that operates independently across multiple data sources [49]. By amalgamating frequent patterns from each source, it aims to uncover patterns relevant across distributed environments. Moreover, the model can be extended to generate rules targeting specific objectives. In [50], authors introduced a novel approach to association rule mining that operates independently across multiple data sources. It merges frequent patterns from each source to unveil patterns applicable across the distributed environment. The method can also be extended to target specific rules. Additionally, it reveals meaningful relationships, enabling access to frequent patterns from individual sources and the entire dataset.

In this work, we introduce RSA-ARM as an advancement in association rule mining (ARM), aiming to address limitations identified in related research. In contrast to existing methods, RSA-ARM leverages the Reptile Search Algorithm (RSA) principles, offering a novel approach to association rule mining. This paper positions RSA-ARM against various existing techniques in the field, highlighting its advantages and contributions to the domain.

4. Proposed Approach

This research introduces a novel RSA-ARM Algorithm rooted in the original Reptile Search Algorithm. This approach aims to unearth association rules while adhering to minimum support and minimum confidence criteria, ensuring reasonable execution times, minimizing computational demands, and preserving the quality of rules. The fundamental steps of this approach are as follows:

A. Encoding Data

Every individual in swarm approaches acts as a potential solution within the explored space. Each member represents a potential rule when addressing the issue of association rule mining (ARM). These individuals are transformed into rules through encoding, making each solution in the swarm a candidate association rule. In the literature, various representations of association rules for mining using genetic algorithms or meta-heuristic algorithms have been proposed. Indeed, when it comes to representing rules to be mined, there are two primary techniques: binary encoding and integer encoding. These methods convert the rules into a format suitable for computational analysis[51]. In binary encoding, n element tables S represent each solution (rule). Within this table, if i is in the rule, S[i] is 1; otherwise, it is 0. This method creates a binary representation of rule items. For example, in a dataset of grocery store transactions where Transaction 1 includes Milk, Bread, and Eggs, and Transaction 2 includes Bread and Butter, the binary vectors would represent these transactions accordingly. Where each position in the vector corresponds to an item in our dataset:

- Transaction 1: [1, 1, 1, 0] (Milk=1, Bread=1, Eggs=1, Butter=0)
- Transaction 2: [0, 1, 0, 1] (Milk=0, Bread=1, Eggs=0, Butter=1)
- Transaction 3: [1, 0, 1, 1] (Milk=1, Bread=0, Eggs=1, Butter=1)

Using binary encoding, suppose we want to represent the association rule $Milk, Bread \rightarrow Eggs$. Representation of the rule is as follows:

- Antecedent (Milk, Bread):[1, 1, 0, 0] (Milk=1, Bread=1, Eggs=0, Butter=0)
- Consequent (Eggs): [0, 0, 1, 0] (Milk=0, Bread=0, Eggs=1, Butter=0).

In contrast, integer encoding uses a vector S to represent the solution X. This vector comprises the k+1 rank, where S[0] is the separator between the rule's previous and subsequent elements. Position k contains value i if the *ith* element is a component of the rule; otherwise, it is set to 0. The index k varies between $0 < k <= n+1$, accommodating the rule's structure. In this work, integer representation is employed. For instance, let $I = \{i1, i2, ..., i10\}$ represent a collection of items:

- $L1 = \{3, 8, 9, 0, 1, 4, 0, 0, 5, 0, 0\}$ represents the rule R1: i8, i9 → i1, i4, i5,
- $L2 = \{5, 0, 2, 3, 7, 1, 0, 0, 0, 0, 8\}$ represents the rule R2: i2, i3, i7, i1 → i8,
- $L3 = \{2, 1, 0, 0, 0, 0, 5, 0, 0, 8, 0\}$ represents the rule R3: i1 → i5, i8,

B. Fitness Function

The presented Algorithm primarily aims to maximize the fitness function, which extracts the most valuable and significant association rules. The objective function is a key used in quality evaluation. As previously explained, the Association Rule Mining issue entails the discovery of every rule that meets minimum support and confidence criteria. In Eq. (12), the fitness function is created with the two empirical parameters α and β:

$$F(r) = \begin{cases} \alpha * \text{Sup}(r) + \beta * \text{Conf}(r) & \text{if accepted} \\ -1 & \text{otherwise} \end{cases} \quad (12)$$

Where α and β are coefficients that control the contribution of support (Sup) and confidence (Conf) to the overall fitness of the rule r, by adjusting these coefficients, we can emphasize the importance of support and confidence or strike a balance between the two.

C. Approach Description

Algorithm 2 shows the RSA-ARM code. It consists of several main steps. First, the "Dataset preprocessing" step in the modified Reptile Search Algorithm is based on a vertical dataset layout. We designed this layout to reduce the computational requirements. Unlike a horizontal dataset layout, where a complete database scan might be necessary, the vertical layout simplifies this process. As a result, the Algorithm can efficiently work with the data to generate and evaluate rules without needing a full database scan. This approach helps optimize the computational resources and improves the Algorithm's performance in rule generation and evaluation for association rule mining (ARM).

The Reptile Search Algorithm (RSA) generates random possible solutions. The Algorithm explores various positions through a series of iterations to identify the optimal solution. Every solution adjusts its position according to the best solution discovered thus far. This process repeats to converge toward optimal solutions. In our proposal, each "reptile" represents a candidate rule that consists of "n" items, with "n" representing the item number within the transactional database. The core idea for generating new rules is to adjust the values of these items within each rule using the RSA approach, which comprises two phases: the hunting and the encircling phases. These phases work together to change the values of the items in each candidate rule to improve how well it solves the target problem, such as association rule mining. By applying the RSA processes, the Algorithm systematically explores and enhances the

Algorithm 2 Reptile Search Algorithm for ARM

```
Require: Max number of iterations $T$, Reptile population, Min support, and Min confidence
 1: Initialize RSA criterion $\alpha$, $\beta$, etc.
 2: Initialize randomly the solutions $X$: $i = 1, \ldots, N$.
 3: while $v < T$ do
 4:     Compute the Fitness for the nominee solutions ($X$).
 5:     Determine the current best solution.
 6:     Modify the $ES$ using Equations (8).
 7:     The start of the RSA.
 8:     for $i = 1$ to $N$ do
 9:         Updating and adjusting the UB and LB.
10:         Generate a new solution $X_i$ using Algorithm 3.
11:         for $j = 1$ to $n$ do
12:             Update the $R$,$P$, and $eta$ using Equations (6), (7), and (9), respectively.
13:             if ($v \leq \frac{T}{4}$) then
14:                 $x_{i,j}(1 + v) = [\beta \times -\eta_{i,j}(v) \times Best_j(v)] -$ rand $\times R_{i,j}(v)$    // High walking
15:             else if ($v \leq \frac{2T}{4}$ and $v > \frac{T}{4}$) then
16:                 $x_{i,j}(1 + v) = ES(v) \times Best_j(v) \times$ rand $\times x_{r_1,j}$    // Belly walking
17:             else if ($v \leq \frac{3T}{4}$ and $v > \frac{2T}{4}$) then
18:                 $x_{i,j}(1 + v) = P_{i,j}(v) \times$ rand $\times$ Best $_j(v)$    // Hunting coordination
19:             else
20:                 $x_{i,j}(1+v) =$ Best $_j(v) - \eta_{i,j}(v) \times \epsilon -$ rand $\times R_{i,j}(v)$    // Hunting cooperation
21:             end if
22:         end for
23:     end for
24:     Calculate the new Maxfitness value for each Reptile agent.
25:     Rank Reptile to the best solution.
26:     $v = v + 1$
27: end while
28: Output: Best solution found, Fitness values for each reptile agent, and set of rules
29: Post-process results and visualization.
```

configuration of these rules to find valuable patterns or associations within the dataset.

The adjusted Reptile Search Algorithm (Algorithm 2) generates new positions (rules), explicitly using Algorithm 3. When creating these new rules, if the average position is less than the current value of an item (Xi), we increase the item's value; otherwise, we decrease it. If the resulting value falls outside the interval [1...N], it is set to 0. The new rule can have duplicate items, and in such cases, we retain only one item randomly while removing the others. Importantly, this process ensures that no invalid rules are generated. Afterward, the Algorithm calculates the fitness value for the candidate rule (represented as a "Reptile" in this context) by replacing the current best solution, denoted as Xi, with the newly generated solution

Algorithm 3 Generate New Solution

```
Require: Rule $X_{i-1}$, UB, LB, $X_i$
 1: Initialize the sum of $X_i$'s positions
 2: Calculate the average value $Sum$ of all elements of $X_i$
 3: if ($LB \geq$ Rule.Length) then
 4:     $LB \leftarrow 0$
 5: end if
 6: for $i = 0$ to UB do
 7:     if $Sum < X_i$ then
 8:         Increment the item at $LB$ by 1
 9:     else
10:         Decrease the item at $LB$ by 1
11:     end if
12:     if ($Item\_at\_LB \leq 0$) or ($Item\_at\_LB >$ Num attributes) then
13:         Set the value of "Item at LB" to 0
14:     end if
15:     if there are duplicate items in the new rule then
16:         Retain one randomly selected item and remove the others
17:     end if
18:     Increment $LB$
19:     if ($LB \geq$ Rule.Length) then
20:         $LB \leftarrow 0$
21:     end if
22: end for
23: Output: Generate new solution $X_i$
```

for the best solution discovered during the same iteration. The search process continues until it reaches the maximum iterations, usually determined by the user or based on the particular needs of the problem. The goal is repeatedly updating and refining the candidate rules to converge toward the most optimal set of rules or patterns within the dataset based on the defined fitness criteria.

D. Complexity of RSA-ARM

Analyzing the complexity of the RSA algorithm for ARM involves evaluating the computational resources it consumes, including CPU time and memory space. Overall, the iteration number (T), the size of the population (N), the item number (n), and the complexity of the operation within the loops influence the computational complexity of RSA for ARM. The worst-case complexity is typically $O(T * N * n)$, but the specific operations can control it in Algorithm 3 and the fitness calculations. It is essential to consider the actual dataset size and problem specifics in practice to assess the Algorithm's efficiency.

In Algorithm 3, the time complexity is $O(UB)$, which indicates the amount of CPU time it requires, while the memory space usage depends on the size of the data structures used to store rule positions and items. Additionally, memory may be allocated for temporary variables and storage of intermediate results during execution. Overall, the Algorithm's resource consumption scales with the upper bound UB and the size of the rule and item sets.

The computational complexity of Algorithm 2 can be expressed as O(T * N * complexity of Algorithm 3).In this case, the overall complexity of RSA-ARM is O(T * N * UB). It is influenced by the number of iterations (T), the population size (N), and the complexity of Algorithm 3, denoted by the upper bound parameter UB.

5. Experimentation Discussion

In our research, we present the experimental outcomes of the RSA-ARM Algorithm with various benchmark dataset sizes (small, medium, and large). Our research methodology involves conducting an initial experiment to determine the optimal parameters for our approach, ensuring its effectiveness. Following this, we position our approach against other algorithms through a comprehensive comparative study, evaluating aspects such as execution time, fitness level, rule mining, support, and confidence. We compare our method with single-objective optimization, exact techniques, and various existing approaches across different dimensions. Finally, we rigorously test the RSA-ARM Algorithm to uncover its limitations and challenges. This coherent approach enables us to systematically analyze our method's performance, strengths, weaknesses, and areas for further improvement.

A. Datasets

To assess the efficacy of our proposed method, we present the experimental outcomes of the RSA-ARM Algorithm with various dataset sizes (small, medium, and large) of different commonly employed in the data mining community. These datasets originate from the Frequent Itemset Mining Dataset Repository [52] and the Bilkent University Function Repository. The second is acquired from the ”LUCS-KDD Discretized/Normalized” database[53]. Table I lists the datasets used. We compare these results with existing algorithms from the literature and newer ones to assess their performance and efficacy.

TABLE I. Dataset Details

Dataset Name	Transactions	Item Size	Description
Basketball	96	5	Small
IBM – Quest	1,000	20	Small
Quake	2,178	4	Small
Chess	3,196	75	Small
Mushroom	8,124	119	Small
BMS-W1	59,602	497	Medium
Connect	100,000	999	Medium
Ecoli	336	34	Small
Breast	699	20	Small
Flare	1,389	39	Small
Led7	3,200	24	Small
BMS-POS	515,597	1,657	Large
WebDocs	1,692,082	526,765	Large

All the algorithms used in this experiment were implemented in Java and executed on a computer equipped with an Intel Core i7 processor, 6 GB of memory, and a Windows 10 operating system.

B. Setting Parameters

In the parameter tuning process, we adjust each parameter's (Number of population and Maximum number of iterations) values to balance the fitness function (rule quality) and CPU run time. We test these values on small, average, and large datasets. CPU time is low for small datasets, but rule quality suffers due to limited exploration. Larger datasets yield better rules but require more CPU time. The aim is to find parameter values that optimize this situation. The study involves four datasets, all with transaction sizes averaging IBM-standard, Quake, Chess, and Mushroom datasets. The RSA-ARM executes 20 times for each dataset, and we average the outcomes.

We established the support threshold at 0.2 and set the confidence threshold at 0.5, with a fixed number of populations of 30 reptiles, alpha and beta equal to 1. This iterative approach allows us to fine-tune the Algorithm's parameters and assess its performance across multiple runs on different datasets. Setting α and β to 1 in the fitness function means we give the same importance or weight to the association rule's support (Sup) and confidence (Conf). With this weightage assigned to both support and confidence, the Algorithm explores the solution space in a balanced manner, considering both the frequency and reliability of the discovered association rules and indicating a balanced approach in evaluating the fitness of association rules, leading to a more comprehensive exploration of the solution space. However, it's essential to carefully analyze the results and consider domain-specific factors to determine the most suitable weightage for support and confidence in practice.

Table II demonstrates that fewer iterations result in shorter execution times due to generating fewer rules. Conversely, increasing the number of iterations results in longer average execution times. This is because more iterations entail developing more rules, requiring more computational time. In summary, the number of iterations directly influences execution time: fewer iterations save time but may limit rule exploration, while more iterations improve exploration but increase execution time. Achieving the right balance is essential for optimal Algorithm performance.

Regarding fitness, our approach achieves its best outcomes with 400 iterations across various datasets. However, the optimal performance was obtained with 100 and 800 iterations for the IBM-Quest and Mushroom. This suggests that the optimal number of iterations may vary depending on the specific dataset, and it is crucial to fine-tune this parameter accordingly to achieve the best results. However, population size is another critical factor that influences the stability and performance of the Algorithm. In the second test, we kept the maximum number of iterations fixed at 400 iterations and systematically varied the number of agents in the population, ranging from 10 to 50. Table III shows the

TABLE II. Evaluating the performance of RSA-ARM regarding the iteration number (Time in Seconds)

#Itr	IBM Quest		Chess		Quake		Mushroom	
	Time	Fitness	Time	Fitness	Time	Fitness	Time	Fitness
100	**0.13**	**1**	0.37	0.68	0.12	0.97	1.62	**0.72**
200	0.24	1	0.66	0.69	**0.21**	**1**	2.20	0.73
300	0.35	1	0.86	0.69	0.32	1	2.35	0.75
400	0.41	1	**1.10**	**0.70**	0.36	1	3.10	0.81
500	0.52	1	1.36	0.70	0.45	1	3.70	0.83
600	0.62	1	1.63	0.70	0.56	1	4.29	0.85
700	0.71	1	1.89	0.70	0.59	1	4.90	0.90
800	0.81	1	2.26	0.70	0.69	1	**5.90**	**0.94**
900	0.86	1	2.26	0.70	0.78	1	6.21	0.94
1000	1.06	1	2.71	0.70	0.83	1	7.0	0.94

Note: The numbers highlighted or bolded in the table represent the best values.

TABLE III. Test the performance of RSA-ARM in terms of the Reptile's population (Time in Seconds)

#pop	IBM Quest		Chess		Quake		Mushroom	
	Time	Fitness	Time	Fitness	Time	Fitness	Time	Fitness
10	**0.15**	**1**	**0.18**	**1**	0.44	0.69	1.10	0.73
20	0.29	1	0.31	1	0.87	0.69	2.26	0.79
30	0.41	1	0.52	1	**1.30**	**0.71**	3.03	0.83
40	0.53	1	0.68	1	1.56	0.71	4.10	0.86
50	0.88	1	1.25	1	1.90	0.71	**5.42**	**0.94**

Note: The numbers highlighted or bolded in the table represent the best values.

outcomes.

Table III demonstrates that employing a few reptiles results in shorter CPU run times. However, this also implies that we explored only a limited portion of the rule space, which can generate lower-quality rules. Increasing the Reptile's number yields better rules but comes at the cost of longer CPU run times.

The results show that optimal fitness in the IBM-Quest, Quake, and Chess datasets is achieved using less than or equal to 30 reptiles. However, in the case of the mushroom dataset, we reached the best fitness with 50 reptiles. This dataset, characterized by its significant number of transactions, is more complex to explore than the others. Hence, we require more reptiles to achieve the best fitness results. These findings underscore the importance of tailoring the number of reptiles to the specific dataset characteristics. Smaller datasets may require fewer reptiles for the efficient exploration, while larger datasets might benefit from a larger reptile population to search the solution space effectively.

The RSA Algorithm is designed to balance exploration, which involves searching a vast solution space to discover potentially better rules, and exploitation, which focuses on refining and optimizing the rules found so far. This delicate balance allows the Algorithm to navigate the search space effectively, discovering high-quality rules while efficiently utilizing computational resources.

C. Performance Comparison

To position our approach effectively versus other algorithms, we have conducted a comparative study that evaluates five key aspects: execution time, fitness level, the mean value of mined rules, mean support, and average confidence. These criteria comprehensively assess the Algorithm's efficiency and the quality of extracted rules. To evaluate effectiveness, this Comparison is divided into two main steps:

1) Comparison with Single-Objective Optimization Approaches and Exact Methods

In this part, we compare the performance of RSA-ARM with several established association rule mining algorithms, including BSO-ARM [8], Pe-ARM [31], MSB-ARM [41], WO-ARM [10], BAT-ARM [38], Apriori [30] and FP-Growth [2]. This comparative analysis allows for a comprehensive assessment of RSA-ARM's effectiveness and efficiency with these well-known algorithms. The parameters employed for these algorithms were determined based on the best values recommended by their respective authors. However, for RSA-ARM, we have established the maximum number of iterations as 400 and the Reptile count as 30.

Table IV presents results obtained by applying RSA-ARM and the other algorithms to various datasets. The primary objective is to maximize fitness as a critical evaluation criterion. The results average over 20 executions for

TABLE IV. RSA-ARM VS Other Approaches W.R.T to Fitness and CPU Time

	Pe-ARM		**BSO-ARM**		**MSB-ARM**		**BAT-ARM**		**WO-ARM**		**RSA-ARM**		**Apriori**	**FP-Growth**
Datasets	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**F**	**T**	**T**
Basketball	1.5	**1**	3.36	0.92	4	**1**	7	0.81	0.7	**1**	0.37	0.96	0.031	**0.016**
IBM Quest	1.68	0.92	1.92	0.93	13	0.84	19	0.41	1.2	0.94	**0.13**	**1**	0.578	0.422
Quake	3.35	0.91	4.5	**1**	40	**1**	76	0.52	2.3	**1**	0.18	**1**	**0.016**	**0.016**
Chess	4.02	0.89	5.1	0.88	13	0.97	141	0.92	4.7	**0.99**	**1.30**	0.71	950	523
Mushroom	10.68	0.88	9.1	0.75	144	0.68	341	0.93	10	**0.97**	**5.42**	**0.97**	2200	1165
BMS-Web-1	323	0.45	370	0.35	296	**0.55**	11352	0	N	N	**93.78**	0.51	1000	800
Connect	870	0.32	950	0.25	971	**0.64**	33521	0	N	N	**194**	0.27	2600	2900

Note: Bolded numbers indicate the best outcomes, and the two columns are for Time (T), Not tested (N), and Fitness (F).

each Algorithm across seven datasets. Overall, RSA-ARM tends to surpass other algorithms in terms of CPU time, with minor exceptions in the Basketball and Quake datasets, where Apriori and FP-growth had slightly shorter runtimes, within a negligible difference of 0.9 seconds.

Regarding fitness value, Pe-ARM, Bat-ARM, and BSO-ARM exhibit relatively lower performance than MSB-ARM, WO-ARM, and the proposed method. Additionally, RSA-ARM demonstrates superiority when working with small datasets containing fewer than 10,000 transactions. For large datasets with over 60,000 transactions, MSB-ARM stands out as the most efficient and outperforms all other methods. These observations underline different algorithms' varying strengths and weaknesses, with MSB-ARM excelling in handling large datasets and RSA-ARM performing well with smaller datasets.

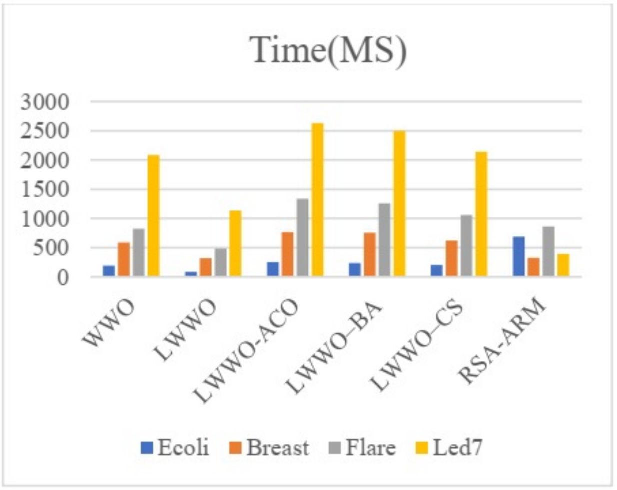


Figure 3. The computational efficiency of different algorithms in terms of execution time

2) *Comparison with other Approaches in Different Aspects*

In this second Comparison, we use average confidence, average support, and average rule mining as supplementary criteria to test on four other benchmark datasets and to assess rule quality as shown in Table V. These metrics provide insights beyond CPU time and fitness, enhancing the comprehensive evaluation of the Algorithm's performance and the practical usefulness of the discovered rules. RSA-ARM's performance compared to several approaches employed in the study, namely LWWO–ACO, LWWO, LWWO–CS, LWWO–BA, and WWO [32]. The parameters utilized for these algorithms were set to the best values recommended by their respective authors. For RSA-ARM, specific parameter values are a maximum of 500 iterations, a population size of 60, α (support weight) set to 0.7, β (confidence weight) set to 0.3, a minimum support of 0.1, and a minimum confidence of 0.5, which aligns with the settings used in other algorithms.

Figure 3 compares the mean execution times associated with the algorithms when applied to various datasets. Notably, as the dataset size of transaction volume increased, the algorithms demonstrated longer execution times. Specifically, the Led7 dataset, which contained more transaction records, exhibited longer average execution times for all algorithms. Across all datasets, the RSA-ARM Algorithm consistently outperformed the LWWO Algorithm in terms of execution time, with improvements ranging from 14.86% to 65.61%. Interestingly, when tested on the Breast and Flare datasets, the three hybrid and the primary WWO algorithms indicated a marginal growth in performance time compared to RSA-ARM. It is worth noting that the LWWO–ACO Algorithm had the longest execution time in this evaluation.

Figure 4 compares the time required by each method and the number of rules produced through each approach for several examples. Remarkably, RSA-ARM outperformed LWWO-BA by a wide margin, producing 6.21–18.68 times the number of rules. This was evident across all datasets. In Comparison to LWWO-BA, the time consumption of RSA-ARM was significantly lower by 31.72% to 84.32%. This indicates that using RSA-ARM has proven to be an effective method for enhancing the diversity of solutions.

Figure 5 compares mean support values among different algorithms. Notably, on the Ecoli dataset, the RSA-ARM Algorithm exhibited a considerably higher mean level of support than the WWO algorithm and all hybrid algorithms. To provide more detail, the support of RSA-

TABLE V. RSA-ARM VS Other Approaches W.R.T to supplementary evaluation criterion

Dataset	Approach	Execution Time (ms)	Rule Mined	Average Support	Average Confidence	Fitness Value
Ecoli	**RSA-ARM**	689	27	**0.457**	0.752	**0.691**
	LWWO	**86**	173	0.325	0.799	0.657
	LWWO–CS	201	371	0.339	**0.826**	0.680
	LWWO-ACO	256	272	0.332	0.816	0.671
	LWWO–BA	237	**513**	0.336	0.815	0.671
	WWO	194	124	0.303	0.792	0.645
Breast	**RSA-ARM**	**275**	**7306**	0.360	**0.884**	**0.710**
	LWWO	323	76	0.332	0.806	0.664
	LWWO–CS	622	376	0.359	0.819	0.681
	LWWO-ACO	769	127	0.346	0.818	0.676
	LWWO–BA	756	391	**0.365**	0.823	0.686
	WWO	586	65	0.307	0.797	0.650
Flare	**RSA-ARM**	861	64	0.168	0.522	0.324
	LWWO	**484**	364	0.156	0.831	0.629
	LWWO–CS	1056	781	**0.193**	**0.930**	**0.709**
	LWWO-ACO	1336	619	0.172	0.863	0.656
	LWWO–BA	1261	**1099**	0.180	0.904	0.687
	WWO	822	252	0.127	0.827	0.617
Led7	**RSA-ARM**	**391**	**5825**	0.136	0.527	0.648
	LWWO	1137	179	0.168	0.837	0.636
	LWWO–CS	2143	896	**0.286**	**0.868**	**0.693**
	LWWO-ACO	2634	271	0.197	0.852	0.656
	LWWO–BA	2495	938	0.224	0.851	0.663
	WWO	2085	155	0.151	0.786	0.596

Note: Highlighted or bolded numbers indicate the best outcomes within their respective group of experiments.

ARM varied from 0.136 to 0.457, LWWO–CS showed variability with support values ranging from 0.193 to 0.359, WWO–ACO displayed a support range between 0.172 and 0.346, LWWO–BA had support values spanning from 0.180 to 0.365, LWWO had support values ranging from 0.156 to 0.332, and finally, WWO's support ranged from 0.127 to 0.307.

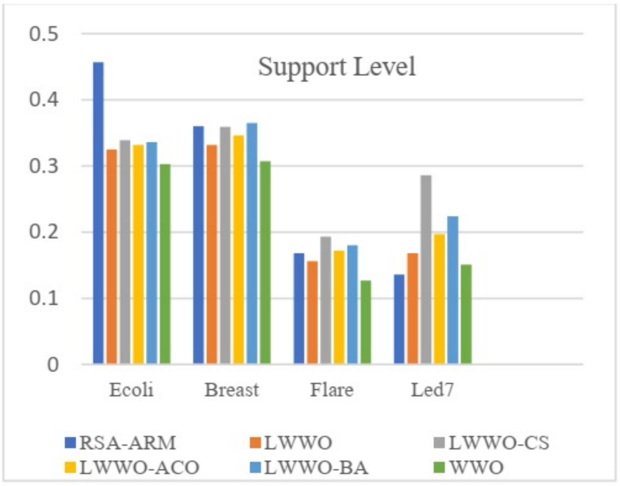


Figure 5. Support average Level of each Algorithm

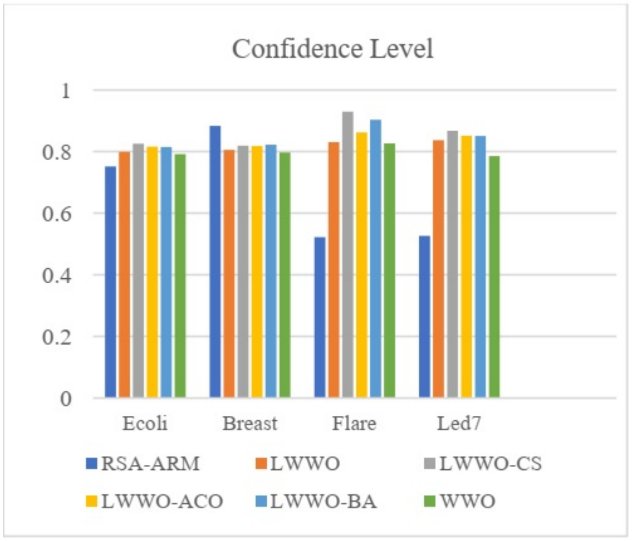


Figure 6. Confidence Average Level Comparison

Figure 6 compares the mean confidence scores for various algorithms applied to four datasets. The results indicate that hybrid algorithms resulted in a subtle improvement in the reliability of the association rules discovered. Notably, among the hybrid algorithms, LWWO–CS showed the most

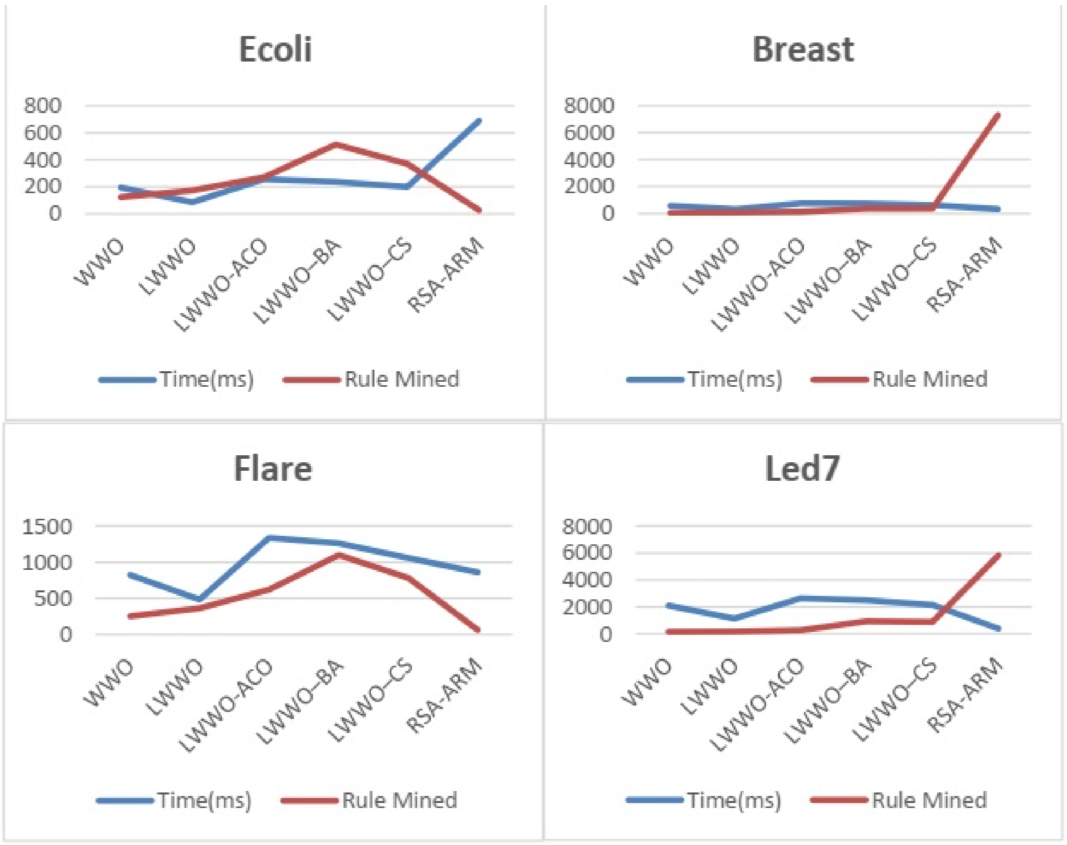


Figure 4. Compares the average CPU time with the mined rules

significant improvement in confidence. For instance, when looking at the Breast dataset, the average confidence of RSA–ARM increased by 7.93% compared to LWWO–CS. This implies that, in this specific dataset, RSA–ARM achieved an increased level of confidence in the ARs it generated, demonstrating the impact of the Algorithm's performance in improving the quality of rules.

Figure 7 compares the mean fitness level across the six approaches. Specifically, RSA-ARM's fitness value ranged from 0.324 to 0.710 across all datasets. On average, the fitness of RSA-ARM improved by 1.67% to 4.25% when compared to LWWO-CS. This indicates that the RSA-ARM algorithm produces association rules of higher quality, and the advantage stands out, especially in the Ecoli and Breast datasets. RSA-ARM performed the best among all the algorithms, indicating its superior fitness and rule quality performance on these datasets.

The RSA-ARM algorithm outperformed other hybrid methods in generating robust association rules with more significant support and confidence. It excelled regarding rule quantity and quality across all datasets. On the other hand, among the hybrid algorithms, LWWO–CS exhibited the quickest processing time. Compared with WWO and the four hybrid algorithms, RSA-ARM notably improved the quantity and quality of mined association rules without significantly increasing the time required. This underscores its effectiveness and reliability in ARM.

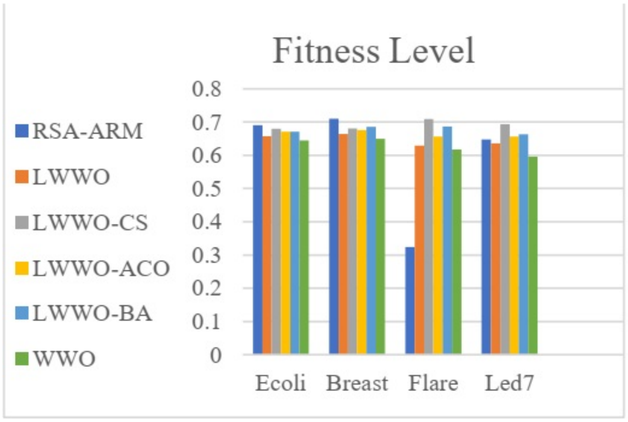


Figure 7. Confidence Average Level Comparison

3) *RSA-ARM Algorithm Limitations*

The experiments described below aim to assess our approach's capabilities and limitations. To do so, we conducted multiple executions on two significant datasets.

Table VI showcases the CPU time required by our proposed method on BMS-POS and WebDocs datasets. Our method demonstrates superior performance compared to improved BSO-ARM and Apriori regarding CPU time on the BMS-POS dataset. However, when applied to the WebDocs dataset, the RSA-ARM algorithm was found to have a complete blockage. This outcome can be attributed to the enormous size or complexity of the transactions within these datasets. Notably, none of the algorithms could provide CPU time results for the WebDocs dataset, indicating the formidable challenges posed by its scale or intricacy in processing within reasonable time constraints. Our research thoroughly evaluated the RSA-ARM algorithm, comparing it with various established ARM algorithms and hybrid association rule mining algorithms with multiple benchmark datasets. Our objective was to comprehensively assess RSA-ARM's effectiveness and efficiency in rule mining.

TABLE VI. Experiments on large datasets in terms of CPU time (sec) of the RSA-ARM

Dataset	RSA-ARM	BSO-ARM	Apriori
BMS-POS	3,254	10,000	22,000
WebDocs	Blocked	Blocked	/

6. Conclusion

Association rules are extensively employed to extract connections between items in databases. However, exact algorithms face challenges in extracting rules due to the significant time and memory required, especially with many transactions and items in the database. This paper introduces a new approach, called RSA-ARM, built upon the Reptile Search Algorithm (RSA) principles for association rule mining. Extensive experiments were performed to assess the performance of the developed methods. Necessary steps were taken to configure the parameters for each Algorithm to their best-performing values, as proposed by the authors, for a subjective and rigorous assessment.

Our study extends beyond traditional metrics like CPU time and fitness by incorporating additional criteria such as average confidence, average support, and average rule mining. These supplementary metrics provide more profound insights into rule quality and practical utility, making our evaluation more comprehensive. The findings of our research demonstrate that RSA-ARM excels across multiple dimensions. It consistently outperforms established algorithms in terms of both effectiveness and efficiency, producing high-quality association rules with superior support and confidence levels. Notably, RSA-ARM showcases its capabilities by achieving the highest number of rules and fitness values across various datasets. Moreover, our study's findings underscore the importance of continuous innovation in association rule mining techniques. While RSA-ARM demonstrates remarkable performance across various datasets, it is crucial to acknowledge its limitations and explore avenues for further refinement. For example, future research could focus on enhancing the Algorithm's scalability to handle even larger datasets or integrating advanced machine learning techniques to automate the parameter tuning process.

We plan to enhance our technique in prospective work to accommodate large-scale datasets effectively. This improvement can be achieved by implementing parallel execution on Graphics Processing Units (GPUs). This approach will enable us to process and analyze vast amounts of data more efficiently, revealing more valuable insights and intricate patterns within large and complex datasets. Ultimately, our study contributes to advancing the field of association rule mining and lays the foundation for future research aimed at harnessing the full potential of data-driven insights for societal and economic benefit.

References

[1] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI Magazine*, vol. 17, no. 3, pp. 37–54, 1996.

[2] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *ACM SIGMOD Record*, vol. 22, no. 2, pp. 207–216, 1993.

[3] R. Agrawal, R. Srikant *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, vol. 1215. Santiago, Chile, 1994, pp. 487–499.

[4] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," *ACM SIGMOD Record*, vol. 29, no. 2, pp. 1–12, 2000.

[5] P. Fournier-Viger, J. C. Lin, B. Vo, T. T. Chi, J. Zhang, and H. B. Le, "A survey of itemset mining," *WIREs Data Mining and Knowledge Discovery*, vol. 7, no. 4, 2017.

[6] J. M. Luna, P. Fournier-Viger, and S. Ventura, "Frequent itemset mining: A 25 years review," *WIREs Data Mining and Knowledge Discovery*, vol. 9, no. 6, 2019.

[7] M. Kaya and R. Alhajj, "Utilizing genetic algorithms to optimize membership functions for fuzzy weighted association rules mining," *Applied Intelligence*, vol. 24, no. 1, pp. 7–15, 2006.

[8] Z. Kou and L. Xi, "Binary particle swarm optimization-based association rule mining for discovering relationships between machine capabilities and product features," *Math Probl Eng*, vol. 2018, pp. 1–16, 2018.

[9] Y. Djenouri, H. Drias, and Z. Habbas, "Bees swarm optimization using multiple strategies for association rule mining," *International Journal of Bio-Inspired Computation*, vol. 6, no. 4, p. 239, 2014.

[10] K. E. Heraguemi, N. Kamel, and H. Drias, "Association rule mining based on bat algorithm," *J Comput Theor Nanosci*, vol. 12, no. 7, pp. 1195–1200, 2015.

[11] H. KamelEddine, H. Kadrii, and A. Zabi, "Whale optimization algorithm for solving association rule mining issue," *International Journal of Computing and Digital Systems*, vol. 10, no. 1, pp. 333–342, 2021.

[12] L. Abualigah, M. A. Elaziz, P. Sumari, Z. W. Geem, and A. H.

Gandomi, "Reptile search algorithm (rsa): A nature-inspired meta-heuristic optimizer," *Expert Syst Appl*, vol. 191, p. 116158, 2022.

[13] Q. Yuan, Y. Zhang, X. Dai, and S. Zhang, "A modified reptile search algorithm for numerical optimization problems," *Comput Intell Neurosci*, vol. 2022, pp. 1–20, 2022.

[14] S. Rajput, R. Chawra, P. S. Wani, and S. J. Nanda, "Noisy sonar image segmentation using reptile search algorithm," in *2022 International Conference on Connected Systems & Intelligence (CSI)*. IEEE, 2022, pp. 1–10.

[15] M. H. Elkholy, M. Elymany, A. Yona, T. Senjyu, H. Takahashi, and M. E. Lotfy, "Experimental validation of an ai-embedded fpga-based real-time smart energy management system using multi-objective reptile search algorithm and gorilla troops optimizer," *Energy Convers Manag*, vol. 282, p. 116860, 2023.

[16] L. H. Son *et al.*, "Arm–amo: An efficient association rule mining algorithm based on animal migration optimization," *Knowl Based Syst*, vol. 154, pp. 68–80, 2018.

[17] M. J. Zaki, "Scalable algorithms for association mining," *IEEE Trans Knowl Data Eng*, vol. 12, no. 3, pp. 372–390, 2000.

[18] T. Uno, T. Asai, Y. Uchida, and H. Arimura, "An efficient algorithm for enumerating closed patterns in transaction databases," in *Discovery Science: 7th International Conference, DS 2004, Padova, Italy, October 2-5, 2004. Proceedings 7*. Springer, 2004, pp. 16–31.

[19] M. Shawkat, M. Badawi, S. El-ghamrawy, R. Arnous, and A. El-desoky, "An optimized fp-growth algorithm for discovery of association rules," *The Journal of Supercomputing*, vol. 78, no. 4, pp. 5479–5506, 2022.

[20] H. Wu, "Data association rules mining method based on improved apriori algorithm," in *Proceedings of the 4th International Conference on Big Data Research*, 2020, pp. 12–17.

[21] Y. Liu and L. Wang, "An optimized association rule data mining algorithm," in *2022 European Conference on Communication Systems (ECCS)*. IEEE, 2022, pp. 10–15.

[22] X. Cai, S. Xu, L. Chen, J. Li, X. Qiu, and B. Zheng, "An improved association rule mining algorithm based on the prior information," in *2022 4th International Conference on Data-driven Optimization of Complex Systems (DOCS)*. IEEE, 2022, pp. 1–7.

[23] S. Zhang, H. Fan, P. Li, and X. Lin, "Association rules algorithm based on dual support and compression matrix," in *2023 IEEE 7th Information Technology and Mechatronics Engineering Conference (ITOEC)*, vol. 7. IEEE, 2023, pp. 1786–1792.

[24] Z. Shu *et al.*, "A modified hybrid rice optimization algorithm for solving 0-1 knapsack problem," *Applied Intelligence*, vol. 52, no. 5, pp. 5751–5769, 2022.

[25] W. Wang and S. M. Bridges, "Genetic algorithm optimization of membership functions for mining fuzzy association rules," 2000, available: http://iris.cs.uml.edu:8080.

[26] R. Haldulakar and J. Agrawal, "Optimization of association rule mining through genetic algorithm," *International Journal on Computer Science and Engineering (IJCSE)*, vol. 3, no. 3, pp. 1252–1259, 2011.

[27] S. Sarkar, A. Lohani, and J. Maiti, "Genetic algorithm-based association rule mining approach towards rule generation of occupational accidents," in *Computational Intelligence, Communications, and Business Analytics: First International Conference, CICBA 2017, Kolkata, India, March 24–25, 2017, Revised Selected Papers, Part II*. Springer, 2017, pp. 517–530.

[28] J. L. Olmo, J. M. Luna, J. R. Romero, and S. Ventura, "Association rule mining using a multi-objective grammar-based ant programming algorithm," in *2011 11th International Conference on Intelligent Systems Design and Applications*. IEEE, 2011, pp. 971–977.

[29] H. Drias, "Genetic algorithm versus memetic algorithm for association rules mining," in *2014 Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC 2014)*. IEEE, 2014, pp. 208–213.

[30] U. Mlakar, M. Zorman, I. Fister, and I. Fister, "Modified binary cuckoo search for association rule mining," *Journal of Intelligent & Fuzzy Systems*, vol. 32, 2017.

[31] S. Krishnamoorthy, G. S. Sadasivam, M. Rajalakshmi, K. Kowsalyaa, and M. Dhivya, "Privacy preserving fuzzy association rule mining in data clusters using particle swarm optimization," *International Journal of Intelligent Information Technologies*, vol. 13, no. 2, p. 1–20, April 2017.

[32] Y. Gheraibia, "Penguins search optimisation algorithm for association rules mining," *Journal of Computing and Information Technology*, vol. 24, no. 2, pp. 165–179, Jun 2016.

[33] K. N. V. D. Sarath and V. Ravi, "Association rule mining using binary particle swarm optimization," *Eng Appl Artif Intell*, vol. 26, no. 8, pp. 1832–1840, Sep 2013.

[34] Y. Djenouri, H. Drias, Z. Habbas, and H. Mosteghanemi, "Bees swarm optimization for web association rule mining," in *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, 2012, p. 142–146.

[35] Y. Djenouri, D. Djenouri, A. Belhadi, P. Fournier-Viger, and J. C.-W. Lin, "A new framework for metaheuristic-based frequent itemset mining," *Applied Intelligence*, vol. 48, no. 12, p. 4775–4791, 2018.

[36] K. E. Heraguemi, N. Kamel, and H. Drias, "Genetic algorithm versus memetic algorithm for association rules mining," in *2014 Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC 2014)*, 2014, p. 208–213.

[37] K. Heraguemi, N. Kamel, and H. Drias, "Multi-population cooperative bat algorithm for association rule mining," in *Proceedings of the Conference Name*, 2015, p. 265–274.

[38] K. E. Heraguemi, N. Kamel, and H. Drias, "Multi-swarm bat algorithm for association rule mining using multiple cooperative strategies," in *Proceedings of the Conference Name*, 2016, p. 1021–1033.

[39] K. Heraguemi, N. Kamel, and H. Drias, "Multi-objective bat algorithm for mining interesting association rules," in *Proceedings of the Conference Name*, 2017, p. 13–23.

[40] Z. Y. et al., "Association rule mining based on hybrid whale optimization algorithm," *International Journal of Data Warehousing and Mining*, vol. 18, no. 1, pp. 1–22, Aug 2022.

[41] Q. H. et al., "Association rule mining through combining hybrid

water wave optimization algorithm with levy flight," *Mathematics*, vol. 11, no. 5, p. 1195, Feb 2023.

[42] H. Moulai and H. Drias, "Association rule mining using new discrete elephant swarm approaches," *Expert Systems*, vol. 40, no. 2, p. e13159, 2023.

[43] Z. Wang, "A multi-level association rule mining algorithm based on nsga-ii for market basket analysis," in *2023 4th Information Communication Technologies Conference (ICTC)*. IEEE, 2023, pp. 294–300.

[44] B. Rokh, H. Mirvaziri, and M. Olyaee, "A new evolutionary optimization based on multi-objective firefly algorithm for mining numerical association rules," *Soft Computing*, pp. 1–14, 2024.

[45] C. Fernandez-Basso, M. D. Ruiz, and M. J. Martin-Bautista, "New spark solutions for distributed frequent itemset and association rule mining algorithms," *Cluster Computing*, pp. 1–18, 2023.

[46] A. Mokkadem, M. Pelletier, and L. Raimbault, "A recursive algorithm for mining association rules," *SN Computer Science*, vol. 3, no. 5, p. 384, 2022.

[47] A. Mokkadem, M. Pelletier, and Raimbault, "Sufrec, an algorithm for mining association rules: Recursivity and task parallelism," *Expert Systems with Applications*, vol. 236, p. 121321, 2024.

[48] B. Siswanto, H. Soeparno, N. F. Sianipar, and W. Budiharto, "Sdfp-growth algorithm as a novelty of association rule mining optimization," *IEEE Access*, 2024.

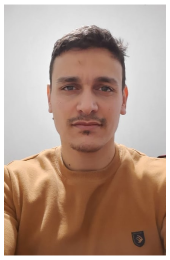
[49] A. Javed and H. Raza, "A novel method for mining association rules independently from multiple data sources," *Soft Computing*, vol. 27, no. 19, pp. 15 959–15 973, 2023.

[50] B. Mudumba and M. F. Kabir, "Mine-first association rule mining: An integration of independent frequent patterns in distributed environments," *Decision Analytics Journal*, p. 100434, 2024.

[51] X. Yan, C. Zhang, and S. Zhang, "Genetic algorithm-based strategy for identifying association rules without specifying actual minimum support," *Expert Systems with Applications*, vol. 36, no. 2, p. 3066–3076, March 2009.

[52] G. B., "Frequent itemset mining dataset repository," 2003, frequent Itemset Mining Implementations (FIMI'03). [Online]. Available: https://cir.nii.ac.jp/crid/1572543025340860416

[53] U. of Liverpool, "LUCS-KDD: Frequent Itemset Mining Implementations (FIMI'03)," https://cgi.csc.liv.ac.uk/frans/KDD/Software/LUCS-KDD, 2003.



Abderrahim Boukhalat is a researcher and Ph.D. student in computer science at Mohamed Boudiaf University of M'Sila, Algeria. He received his license degree in 2016 and a Master's in Computer Science from the same University in 2018. His research interests include data mining, artificial intelligence, and evolutionary computing.

KamelEddine Heraguemi is the Director of Networks and Digital Development at the Ministry of Higher Education and Scientific Research and an Assistant Professor at the National School of Artificial Intelligence, Sidi Abdallah, Algeria. He received his PhD in Computer Science from the University Ferhat Abbas of Setif1 (UFAS1), Setif, and his Master's in Computer Science from the Mohamed-Cherif Messaadia University, Souk-Ahras, Algeria, in 2017 and 2012, respectively. His research interests include data mining, artificial intelligence, and evolutionary computing.

Mohamed Benouis is a computer researcher specializing in machine learning and data mining. He holds a Bachelor's degree in Computer Science from the University of Science and Technology Houari Boumediene, Algiers, Algeria. Currently pursuing his PhD, Mohamed is dedicated to advancing knowledge in artificial intelligence and developing innovative algorithms to solve real-world problems. With a strong focus on research, Mohamed's contributions to the field are highly regarded, and he actively mentors aspiring researchers in computer science.

Brahim Bouderah is an accomplished academic and administrator. With a Ph.D. from the University of U.F.A. Sétif, he has made significant research contributions, particularly in Computing and Mathematics. Currently serving as the Rector of the University Abdelhamid Ibn Badis Mostaganem, Bouderah has demonstrated exceptional leadership and expertise in academic and administrative roles.

Samir Akrouf is a renowned researcher in the field of computer science, with a focus on software engineering and distributed systems. He obtained his Ph.D. in Computer Science from the University of Sciences and Technology Houari Boumediene in Algiers, Algeria. He has contributed significantly to developing efficient algorithms and methodologies for distributed computing and software development.