School of Informatics

Informatics Research Review
The Advantage and Improvement Methods of Genetic Algorithms in Association Rules Mining

[redacted]
January 2021

Abstract

As the amount of available data increases, researchers are trying to come up with new methods to extract useful knowledge. One of the main methods popular in this field is Association rule mining (ARM). Unlike traditional association rule mining algorithms, genetic algorithm-based association rule mining does not require preset minimum support and confidence [1] and has been proven to generate more accurate results. In this article, we will introduce the basic knowledge of GA-based ARM, its advantages and disadvantages compared with other algorithms, and major improvements based on improving the process stages of GA.

Date: Thursday 28th January, 2021

Supervisor: [redacted]

1 Introduction

The recent tremendous growth in the amount of data has driven the advance of data mining techniques for extracting useful information from databases. Data mining is the process of extracting useful knowledge from large amounts of data. The advance of information technology in all areas of real-world life has led to the generation of large amounts of data. Various formats, such as records, images and documents, are used to store all kinds of data. The collected data requires appropriate mechanisms for extracting knowledge from large repositories in order to make better decisions [2]. One of the most popular approach in this field is association rule mining (ARM). The first association rule mining algorithm was proposed in 1993 by Agrawal to extract multi-occurrence patterns and rules from databases [3]. However, the researchers identified some significant shortcomings of Apriori to be addressed. One of these is the massive number of candidate frequent item sets generated. Many researchers worked to reduce this number [4]. Besides, Apriori has another drawback that it needs the user to manually input the preset parameter values, such as minimum confidence and support.

Unlike traditional association rule mining algorithms like Apriori, genetic-algorithms-based association rule mining does not require preset minimum support and confidence values, and has been proved to produce more accurate results [1].

Besides classical ARM algorithms such as Apriori and GA-based approach, there are many other efficient algorithms like other heuristics-based approaches. It is important to know the difference between these algorithms so that proper algorithm can be applied on different problems. This paper will discuss the difference of performance between classical ARM algorithms, heuristics-based ARM algorithms and GA-based approaches.

The first GA-based ARM algorithm was proposed by Yan in 2009, and has been being improved all along these years [1]. A well known proverb states that ”know the enemy and know yourself, and you can fight a hundred battles with no danger of defeat”. It is important to know how these improvements are generated and how to produce further improvement. To achieve this goal, this paper concludes how the GA-based ARM algorithms are improved. Genetic algorithm is of composition of encoding chromosomes, initialization of the population, calculation of the fitness value, selection, mutation, crossover and stop condition [5] [1]. This work focus on the improvements based on components of GA.

To the best of my knowledge, there is no attempt on reviewing the GA-based ARM algorithms at the view of different improvement methods. Luckily, works on the GA-based ARM algorithms itself are sufficient and enough to extract the pattern of how to improve a GA-based approach. This paper will only focus on the improvements on the process of GA, including the initialization, selection, crossover and mutation stage, and the fitness function. Any improvements based on the other aspects are also important for considering improving the algorithm, but will not be discussed in this paper.

In section 2, the basic knowledge of ARM, heuristics and GA is introduced to help to understand following discussing. Comparisons between classical ARM algorithms, heuristics-based ARM algorithms and GA-based approaches are made in section 3.1 with discussion of the specified performance metrics. In section 3.2, the improvements of GA-based ARM algorithms are presented grouped by the process stage on which the improvement based. Later, in section 4, some critical thinking about some drawbacks of GA-based approaches and ideas of further work are present.

2 ARM, Heuristics and GA-based Approaches

This section introduces necessarily important concepts of ARM algorithms, heuristics-based ARM approaches, and GA-based ARM algorithms.

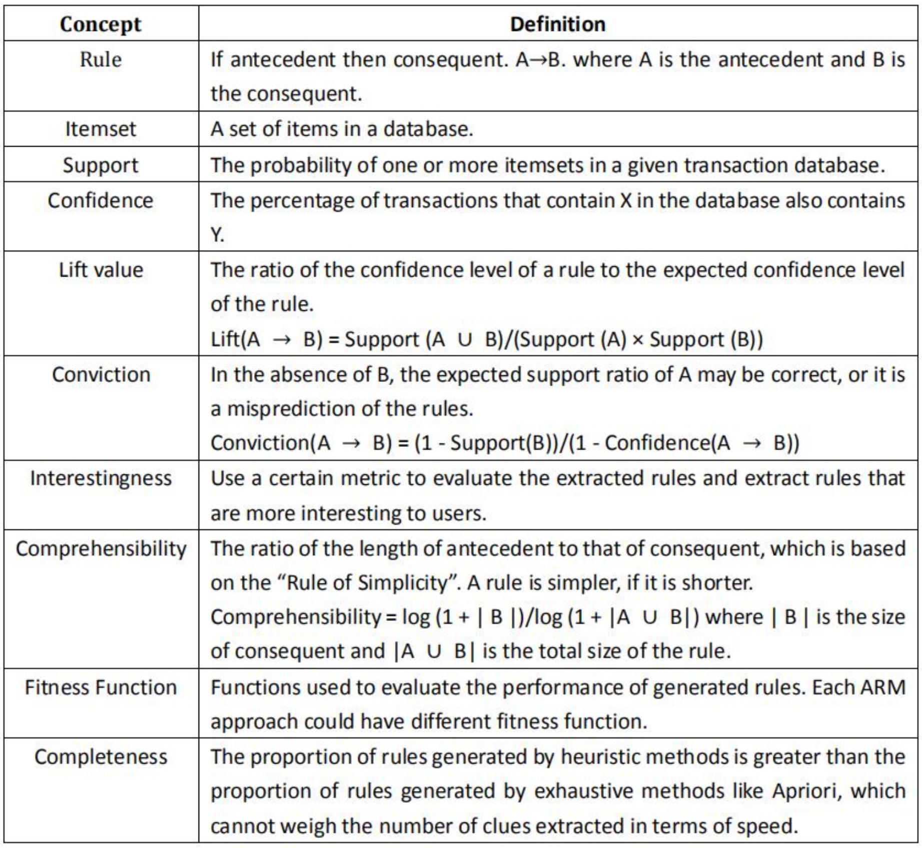
2.1 ARM

This subsection introduces necessarily important key concepts of ARM algorithms.

The first ARM algorithm is proposed by Agrawal, called Apriori [3]. $I = I_1, I_2, ..., I_n$ is a n-size distinct attributes set. T is a transaction composed of a set of items (T $\subseteq$ I), and different Ts constitute a database D. An association rule is an implication that X $\Rightarrow$ Y, where X, Y $\subset$ I are called itemsets, which are different sets of items , and X $\cap$ Y = $\emptyset$. The rule is that X implies Y, where X is called the antecedent while Y is called the consequent,[6].

Support and confidence are two important basic figures for selecting interesting rules from the set space with all possible rules. Usually users will predefined the thresholds for support and confidence to drop those less interesting rules, and the two thresholds are called minimum support and minimum confidence, respectively. Strong rules are these which meets the both requirements of the minimum support and a minimum confidence [6].

In the ARM field there are some necessarily-known concepts and measures as shown in Figure 1.



Concept	Definition
Rule	If antecedent then consequent. A→B. where A is the antecedent and B is the consequent.
Itemset	A set of items in a database.
Support	The probability of one or more itemsets in a given transaction database.
Confidence	The percentage of transactions that contain X in the database also contains Y.
Lift value	The ratio of the confidence level of a rule to the expected confidence level of the rule. Lift(A → B) = Support (A ∪ B)/(Support (A) × Support (B))
Conviction	In the absence of B, the expected support ratio of A may be correct, or it is a misprediction of the rules. Conviction(A → B) = (1 - Support(B))/(1 - Confidence(A → B))
Interestingness	Use a certain metric to evaluate the extracted rules and extract rules that are more interesting to users.
Comprehensibility	The ratio of the length of antecedent to that of consequent, which is based on the “Rule of Simplicity”. A rule is simpler, if it is shorter. Comprehensibility = log (1 + | B |)/log (1 + |A ∪ B|) where | B | is the size of consequent and |A ∪ B| is the total size of the rule.
Fitness Function	Functions used to evaluate the performance of generated rules. Each ARM approach could have different fitness function.
Completeness	The proportion of rules generated by heuristic methods is greater than the proportion of rules generated by exhaustive methods like Apriori, which cannot weigh the number of clues extracted in terms of speed.

Figure 1: List of concepts and measures that are necessarily-known in ARM [7] [8]

2.2 Heuristics approaches

Heuristics represent strategies that use easily available information to control the problem-solving process in humans and machines. Heuristic function (also referred to as a heuristic for short) is a function that ranks the alternatives in the process of search algorithm for each branch step based on the already-got information to determine the branch to follow [9].

The most used heuristics are perhaps EAs [10]. EA is a population-based algorithm, in many cases inspired by biology, it solves problems by simulating the evolution process, which attempts to improve the population(solutions) by making it evolve for several generations. The general scheme of EA is as follows [11]:

1. Initialize the first generation of solutions randomly and forms the first population.
2. Evaluate the performance of each individual of population.
3. Repeat follows until a stop condition is met.
    (a) Select best individuals from population.
    (b) Reassemble the features of individuals of population by different dimensions in order to obtain new individuals, which are called offsprings.
    (c) Evaluate the performance of new individuals.
    (d) Use best offsprings to replace some or all of the individuals of population.
4. Return the individual with best performance so far.

Common heuristics, especially EA, were not originally designed for learning, but have been widely used in data resolution over the past few years due to their successful solution of other combination / numerical optimization problems. It came to be used to delve into the problem. In fact, using heuristics-based algorithms in data mining-based issues is a popular research topic today. [11]

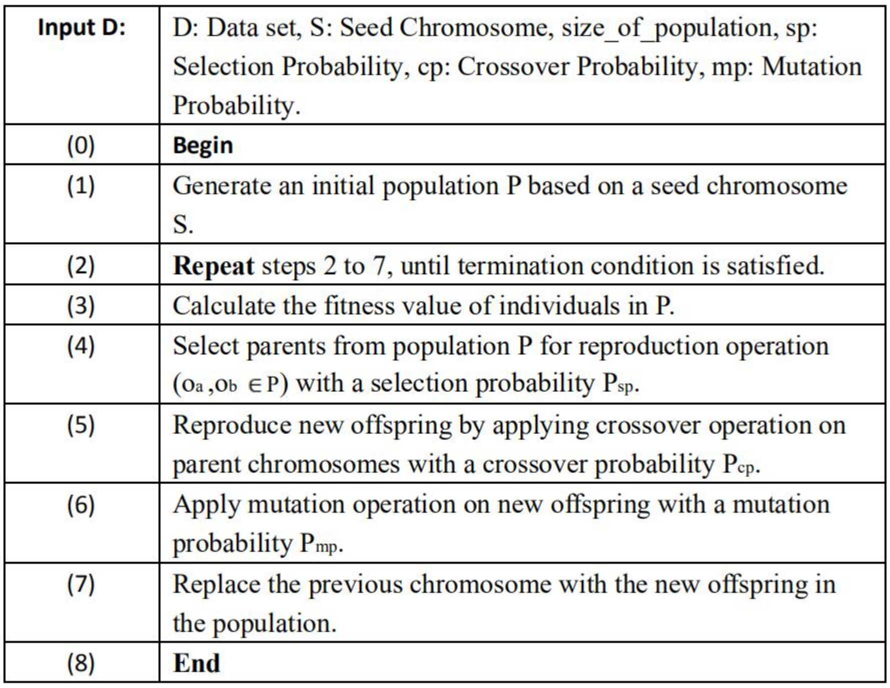
2.3 GA approaches for ARM

2.3.1 GA

The Genetic Algorithm (GA) was proposed by John Holland in 1970. GA is a natural-selection and natural-genetics based random search algorithm, and has been used in some ML practices and optimization problems and produces some delightful solutions. The genetic algorithm works repeatedly by generating new string padding from old strings. The genetic algorithm is inspired by Darwin's evolution theory [12].

The algorithm begins with a randomly generated group of different individuals and is passed down from generation to generation. In each iteration, we evaluate the adaptability of each individual of the population, select the relatively appropriate individual from the old population, and apply genetic operators, normally including selection, crossover and mutation, to generate a new population. Subsequently, the generated population generates new population and be replaced in the next iteration of the algorithm. The algorithm stops when the iteration time meets a preset value or the group reaches a sufficient level of adaptation. The process of a simple GA is shown in Figure 2. The standard GA uses three genetic operators: selection, crossover, and mutation. The functions of the genetic operator are as follows [8]:

- Selection: Selection acts as a filter of chromosome and deals with the probability of survival of the fittest, and choose suitable individuals to stay in population. There are many ways to choose the best chromosome, such as rank selection, roulette wheel selection, tournament selection, etc. [13]
- Crossover: Use two-point strategy to obtain single chromosomes from parents and merge them to form new chromosomes. Crossovers include single point crossovers and multipoint crossovers.
- Mutation: By considering mutation probability (pm) and fitness value at the same time, the chromosome can be changed by changing the boundary between the previous attribute and the next attribute in the same rule. In addition, the operator can randomly select genes and change the attribute index and interval associated with them. Please note that the new interval interval is always the union of the basic intervals, and these basic intervals constitute the subdomain of the new attribute.



Input D:	D: Data set, S: Seed Chromosome, size_of_population, sp: Selection Probability, cp: Crossover Probability, mp: Mutation Probability.
(0)	**Begin**
(1)	Generate an initial population P based on a seed chromosome S.
(2)	**Repeat** steps 2 to 7, until termination condition is satisfied.
(3)	Calculate the fitness value of individuals in P.
(4)	Select parents from population P for reproduction operation ($o_a, o_b \in P$) with a selection probability P_{sp}.
(5)	Reproduce new offspring by applying crossover operation on parent chromosomes with a crossover probability P_{cp}.
(6)	Apply mutation operation on new offspring with a mutation probability P_{mp}.
(7)	Replace the previous chromosome with the new offspring in the population.
(8)	**End**

Figure 2: Simple genetic algorithm [14]

2.3.2 GA used in ARM

This subsection will only introduce the first GA-based ARM algorithm called ARMGA since the underlying of GA-based ARM algorithms are basically not of much difference.

Yan [1] proposed a new method of GA-based ARM algorithm. In their approach, Instead of using preset minimum support threshold, they use a dynamic fitness function of relative minimum confidence and only select the best rule. The old population is pop [i]. First we can apply the selection operator on pop[i] to generate a new population pop[i+1] and then use crossover function to produce new generation by reproducing two offspring from two random chromosomes for each individuals of population. Finally each produced chromosome gets mutated at a probability of mp. After sufficient number of loops, algorithm ARMGA will output a group with high-quality chromosomes.

Algorithm ARMGA stops reproducing the offspring and population, if and only if one of the following cases happens:

1. The difference between individual with best performance and the one with worst performance is less than a preset value, which is always very small.
2. The iteration time already goes beyond the preset maximum number of loops.

3 Literature Review

This section discusses the performance and improvements methods of GA-based ARM algorithms.

3.1 Performance of GA in ARM

In this subsection, the performance of GA-based ARM algorithms will be discussed by comparing to classical and heuristics-based ARM algorithms.

3.1.1 Compared to classical ARM

To compare the effiency and effectiveness of GA-based ARM algorithms and classical ARM algorithms, Jesus [15] used two read-world databases: Stulong and House_16H to apply these methods and analyze the performance. For GA-based methods, they took EARMGA [1], GAR [16] and GENAR [17] into consider. For classical algorithms, they chose to use Apriori [18] and Eclat [19] as standard.

Based on their systematically experiment results, they obtained three facts.

1. The extraction methods based on genetic association rule can obtain high-confidence and good-converge association rules of the database and provide high quality rules to the users.
2. Although the population size of the method will restricts the rules size , the extraction methods based on genetic association rule can reduce the set size of the association rules, and these rules take less attributes in the antecedent into consideration, which is helpful for easier understanding for users.
3. The runtime scales mostly linearly when increasing the size of the problem for the genetic association rule extraction methods.

Seyed Mohssen Ghafari [7] compared several variants of GA-based ARM algorithms such as ARMGA, AGA [20] and IARMGA [21]), and classic ARM algorithms including Apriori and FP-Growth [4] using eight performance metrics, as shown in Figure 3.

1. For execution time. Guo and Zhou [22] stated that improved GA-based ARM algorithm is 1.5 times faster than Apriori.

2. For interestingness. Many people have developed ARM methods to mine the more interesting association rules, such as Yan [1] and Minaei [23] proposed improved GA-based ARM methods with new defined interestingness.

3. For automatic procedure. The compulsory manual setting of minimum support and confidence is one of the biggest drawback of classical ARM approaches, because these values could strongly affect the performance of the ARM algorithms, especially in massive databases. But some researchers noticed this shortcoming and proposed some ARM algorithms such as GA-based methods [24] to determine the minimum support and confidence automatically.

4. For completeness. Djenouri declared that ARMGA may generate rules that is lower than minimum support threshold and minimum confidence threshold and may even produce false rules [25]. Drias claimed that the new approach call IARMGA they proposed solved the generating false rules problem [21].

5. For number of database scans. No significant difference between classical and GA-based approaches.

6. For parallel processing. Among papers collected by Seyed [7], only Dash [26] and Djenouri [27] proposed ARM approaches based on a parallel structure, both of which are GA-based.

7. For the size of generated rules. To decrease the time consumption and needed memory, ARM researchers have tried to reduce the number of generated rules and produce only the most interesting rules. ARMGA generate 1.11 times less rules than Apriori. [1].

8. For memory usage. It seems no GA-based approach has been analyzed for this area.

Based on above work on comparison of classical and GA-based ARM algorithms, it is clear that GA-based approach is better than classical one, especially in terms of execution time, automatic procedure and parallel processing.

However, some work claimed that most of bio-inspired algorithms have two main drawbacks: Firstly, they may generate rules that is lower than minimum support threshold or confidence threshold and may even produce false rules. Secondly, some inadmissible solutions may be generated during the process and there is no treatment address this issue.[21]

3.1.2 Compared to other heuristics-based ARM

Seyed compared all kinds of current ARM algorithms using eight performance metrics, including exhaustive ARM approaches and heuristics-based ARM algorithms [7], as shown in Figure 3. The results are as following.

1. For execution time. The first execution time of AGA (54.75s) [20] and IARMGA [21] (60.25s) is less than BSO-ARM (82.25s) [25] and HSBO-TS [28] (87.54s), but much higher than PMES (0.301s) [27].

2. For interestingness. Many people have developed ARM methods to mine the more interesting association rules, including GA-based and heuristics-based ARM algorithms.

3. For automatic procedure. Researchers have proposed ARM algorithms with automatic procedure based on GA and other heuristics.

4. For completeness. The order of approaches with highest fitnee value is HSBO-TS with 13.39 [28], BSO-ARM with 11.14 [25], IARMGA with 11.03 [21], ARMGA with 8.37 [1], AGA with 3.37 [20].

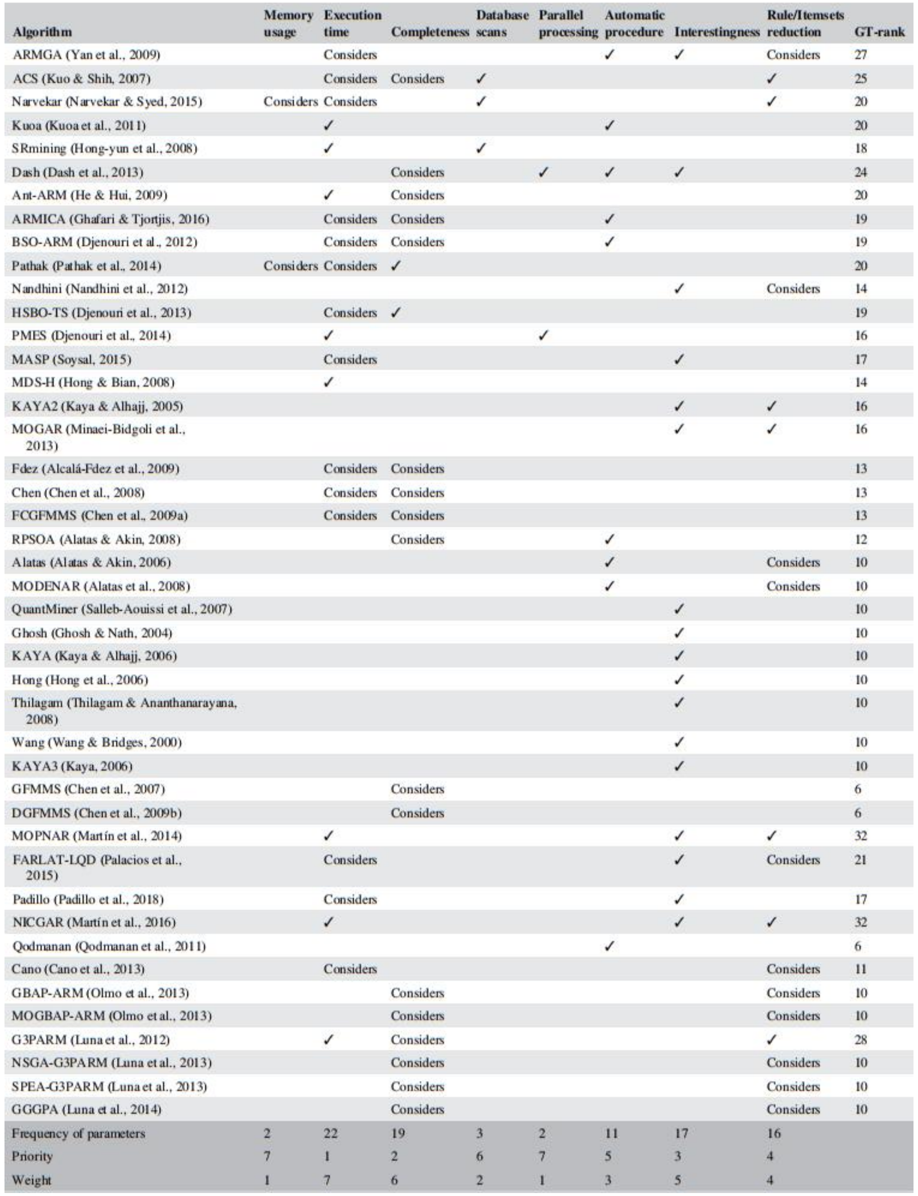
5. For number of database scans. GA-based approach is inferior compared some other heuristics-based ARM algorithms requiring less database scans.

6. For parallel processing. Among papers collected by Seyed [7], only Dash [26] and Djenouri [27] proposed ARM approaches based on a parallel structure, both of which are GA-based.

7. For size of generated rules. ACS produces about twenty-three times less rules compared to Apriori, while the figure for ARMGA is only 1.11 [29] [1].

8. For memory usage. No experimental results on GA-based approaches.

Based on above comparison and a parameter called GT-Rank , Seyed claimed that ARMGA, ASC, and Kua emerge as the best ARM approaches [7].



Algorithm	Memory usage	Execution time	Completeness	Database scans	Parallel processing	Automatic procedure	Interestingness	Rule/Itemsets reduction	GT-rank
ARMGA (Yan et al., 2009)		Considers				✓	✓	Considers	27
ACS (Kuo & Shih, 2007)		Considers	Considers	✓				✓	25
Narvekar (Narvekar & Syed, 2015)	Considers	Considers		✓				✓	20
Kuoa (Kuoa et al., 2011)		✓				✓			20
SRmining (Hong-yun et al., 2008)		✓		✓					18
Dash (Dash et al., 2013)			Considers		✓	✓	✓		24
Ant-ARM (He & Hui, 2009)		✓	Considers						20
ARMICA (Ghafari & Tjortjis, 2016)		Considers	Considers			✓			19
BSO-ARM (Djenouri et al., 2012)		Considers	Considers			✓			19
Pathak (Pathak et al., 2014)	Considers	Considers	✓						20
Nandhini (Nandhini et al., 2012)							✓	Considers	14
HSBO-TS (Djenouri et al., 2013)		Considers	✓						19
PMES (Djenouri et al., 2014)		✓			✓				16
MASP (Soysal, 2015)		Considers					✓		17
MDS-H (Hong & Bian, 2008)		✓							14
KAYA2 (Kaya & Alhajj, 2005)							✓	✓	16
MOGAR (Minaei-Bidgoli et al., 2013)							✓	✓	16
Fdez (Alcalá-Fdez et al., 2009)		Considers	Considers						13
Chen (Chen et al., 2008)		Considers	Considers						13
FCGFMMS (Chen et al., 2009a)		Considers	Considers						13
RPSOA (Alatas & Akin, 2008)			Considers			✓			12
Alatas (Alatas & Akin, 2006)						✓		Considers	10
MODENAR (Alatas et al., 2008)						✓		Considers	10
QuantMiner (Salleb-Aouissi et al., 2007)							✓		10
Ghosh (Ghosh & Nath, 2004)							✓		10
KAYA (Kaya & Alhajj, 2006)							✓		10
Hong (Hong et al., 2006)							✓		10
Thilagam (Thilagam & Ananthanarayana, 2008)							✓		10
Wang (Wang & Bridges, 2000)							✓		10
KAYA3 (Kaya, 2006)							✓		10
GFMMS (Chen et al., 2007)			Considers						6
DGFMMS (Chen et al., 2009b)			Considers						6
MOPNAR (Martín et al., 2014)		✓					✓	✓	32
FARLAT-LQD (Palacios et al., 2015)		Considers					✓	Considers	21
Padillo (Padillo et al., 2018)		Considers					✓		17
NICGAR (Martín et al., 2016)		✓					✓	✓	32
Qodmanan (Qodmanan et al., 2011)						✓			6
Cano (Cano et al., 2013)		Considers						Considers	11
GBAP-ARM (Olmo et al., 2013)			Considers					Considers	10
MOGBAP-ARM (Olmo et al., 2013)			Considers					Considers	10
G3PARM (Luna et al., 2012)		✓	Considers					✓	28
NSGA-G3PARM (Luna et al., 2013)			Considers					Considers	10
SPEA-G3PARM (Luna et al., 2013)			Considers					Considers	10
GGGPA (Luna et al., 2014)			Considers					Considers	10
Frequency of parameters	2	22	19	3	2	11	17	16	
Priority	7	1	2	6	7	5	3	4	
Weight	1	7	6	2	1	3	5	4	

Figure 3: List of the ARM algorithms with corresponding eight performance metrics [7]

3.2 Improvements in GA-based ARM algorithms

In this subsection, the improvements of GA-based ARM algorithms are presented grouped by the process stage on which the improvement based, including initialization stage, crossover stage, mutation stage, selection stage and the fitness function.

3.2.1 Improvements in initialization stage

Some researches have shown the non-negligible influence from the initialization of population on the produced solution [30]. Traditional GA-based approaches initialize the population using a single seed, of which the main idea is to select a set of chromosomes from a massive solution space randomly, which is called the seed chromosome. However, some researchers have found that it is not guaranteed that the following generation will traverses the whole solution space if the generation of a initialized population is based on a single seed.

To avoid these problems and in order to discover rules of high quality from large datasets, Kabir [14] proposed a new multi-seed-based genetic algorithm, MSGA, which can generate plural seeds from different space areas of the set of possible solution. The architecture of the MSGA is shown in Figure 5. They took experiments on four real-world datasets, and stated that most of the time the MSGA tends to get to a convergence faster than methods based on single seeds and tends to presents better or similar high-quality rules.

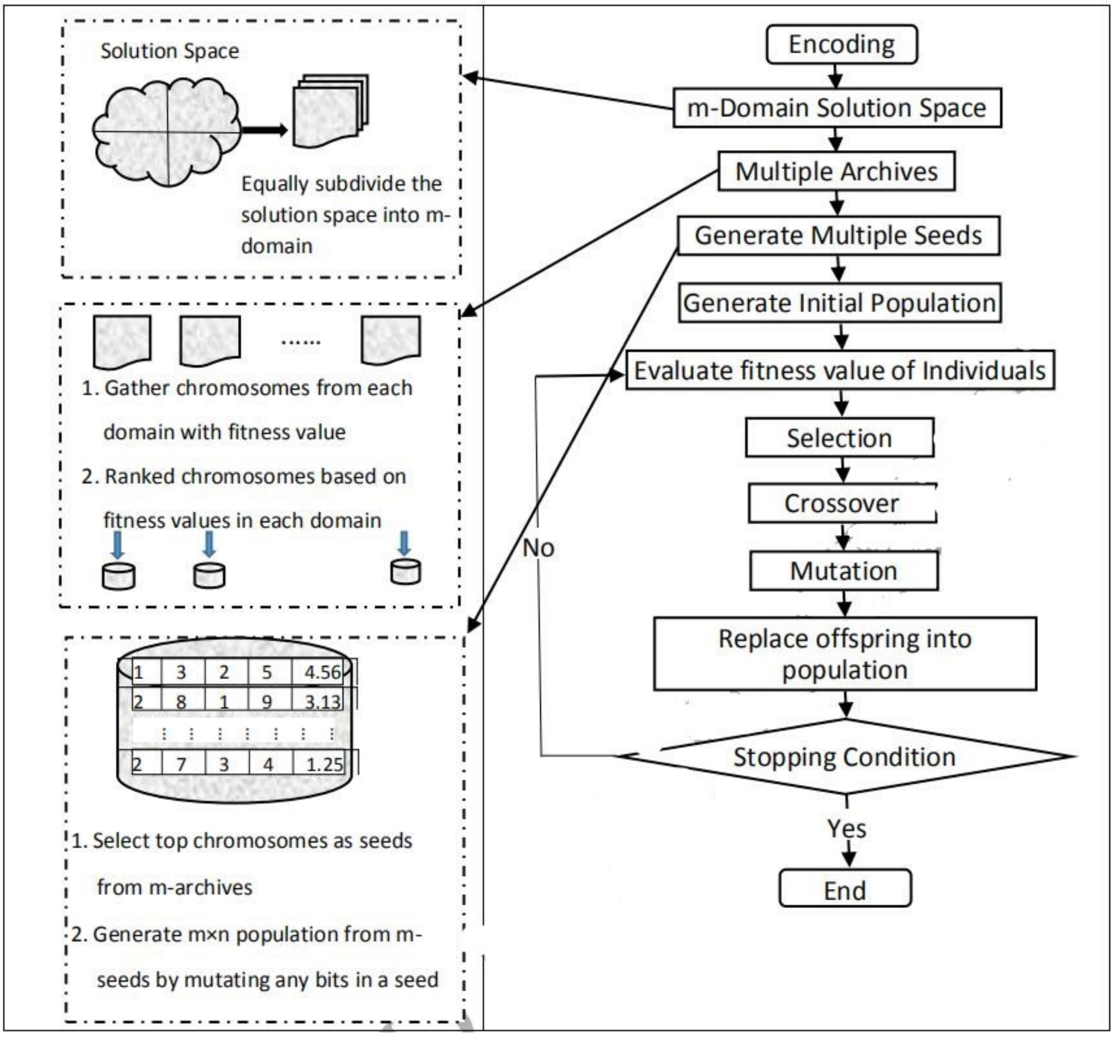


Figure 4: The architecture of the MSGA [14]

3.2.2 Improvements in crossover stage

Wang, Zou, and Liu [20] proposed a new adaptive GA-based ARM algorithm called AGA that uses mutation matrices and crossover matrices for multidimensional ARMs. The crossover part idea behind AGA is that crossover likelihood is a dynamic function of distance matrix defined by a distance measure between two chromosomes. A chromosome with smaller distance matrix has higher likelihood to be selected as the first chromosome for crossover. The second chromosome is chosen by the possibility depended on the chromosome already chosen.

But Djenouri studied AGA and then he claimed that there is a significant drawback of AGA. The AGA generates some false rules and fails compared to his proposed method in terms of fitness level and consummated time [28].

3.2.3 Improvements in mutation stage

The mutation part idea behind AGA is that mutation probability is a preset function of time, fitness ranking, and locus. Chromosomes with higher fitness value have a lower likelihood of being involved in mutation and fewer loci to undergo mutation [20].

Kabir [31] also applies adaptive mutation on the GA-based approach, called ARMGAAM. Genetic algorithms with ranked mutation perform better than these with fixed mutation [32]. During the process of ARMGAAM, the likelihood of mutation is decreasing along with the increasing of the iteration times. Mutation rate is also adapted depending on the fitness value of different individuals of a generation. The possibility of a offspring with topper rank getting mutated is lower than that with lower rank.

3.2.4 Improvements in fitness function and selection stage

Minaei-Bidgoli [23] proposed a new multi-objective GA-based ARM algorithm for numerical ARM based on the notion of rough patterns. Pareto is used to calculate the fitness value of individuals/rule of a generation. The chromosomes with the smallest rank score get the highest fitness value and these with equivalent rank score get the same fitness value [33].

They analyzed the performances of the multi-objective GA-based approach on three real-world datasets, two of which are overwhelmingly better in terms of number of produced rules. Average improvements are 50 percentage and 32 percentage respectively. The support and size measures are almost the same.

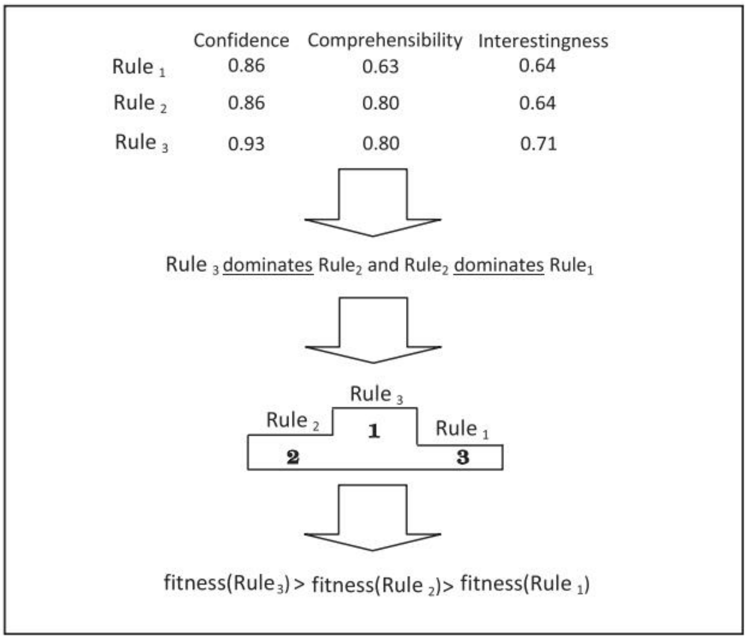


Figure 5: Calculating the fitness of three generated rules. [23]

4 Current Issue and Future Directions

During this paper is being conducted, some research gaps have been found.

1. Many association rule mining algoriehms are advanced for datasets with non-large scale, and thus the performances of algorithms on large datasets are sometimes neglected, which is non-neglect for performance analysis.
2. The Apriori may not be a proper standard to analyze the performance of a ARM algorithm, because it requires many intermediate process to calculate all occurance times of patterns and then has computational complexity and memory consumption [34]. A recently proposed ARM approaches with better performance would be better.
3. Memory consumption, as discussed in section 3.2, is one of the main performance metrics to analyze the performance of ARM [35], but only few researchers consider memory consumption as an evaluation parameter [7].

5 Summary & Conclusion

The reviewed papers showed the excellence of performance of GA-based ARM algorithms compared to classical algorithms and other heuristics-based approaches, especially in terms of rule/itemsets reduction, automatic procedure and parallel processing, although there are some nonnegligible drawbacks to be addressed. And some important works have been discussed and grouped by the improvement methods, including initialization, crossover, mutation and selection. Finally, some suggests on further research are present.

References

[1] Xiaowei Yan, Chengqi Zhang, and Shichao Zhang. Genetic algorithm-based strategy for identifying association rules without specifying actual minimum support. *Expert Systems with Applications*, 36(2 PART 2):3066–3076, 2009.

[2] Venkatadri. M and Lokanatha C. Reddy. A Review on Data mining from Past to the Future. *International Journal of Computer Applications*, 15(7):19–22, 2011.

[3] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining Association Rules Between Sets of Items in Large Databases. *ACM SIGMOD Record*, 22(2):207–216, 1993.

[4] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 29(2):1–12, 2000.

[5] Murat Albayrak and Novruz Allahverdi. Development a new mutation operator to solve the traveling salesman problem by aid of genetic algorithms. *Expert Systems with Applications*, 38(3):1313–1320, 2011.

[6] Sotiris Kotsiantis and Dimitris Kanellopoulos. Association Rules Mining: A Recent Overview. *Science*, 32(1):71–82, 2006.

[7] Seyed Mohssen Ghafari and Christos Tjortjis. A survey on association rules mining using heuristics. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(4):1–29, 2019.

[8] Gagandeep Kaur and Shruti Aggarwal. A Survey of Genetic Algorithm for Association Rule Mining. *International Journal of Computer Applications*, 67(20):25–28, 2013.

[9] Judea Pearl. *Heuristics: intelligent search strategies for computer problem solving.* 1984.

[10] J E Smith and A.E. Eiben. *Introduction toEvolutionaryComputing.* 2015.

[11] María J. del Jesus, José A. Gámez, Pedro González, and José M. Puerta. On the discovery of association rules by means of evolutionary algorithms. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(5):397–415, 2011.

[12] Dimple S.Kanani and Shailendra K. Mishra. An Optimized Association Rule mining using Genetic Algorithm. *International Journal of Computer Applications*, 119(14):11–15, 2015.

[13] J Malar Vizhi. Data Quality Measurement on Categorical Data Using Genetic Algorithm. *International Journal of Data Mining & Knowledge Management Process*, 2(1):33–42, 2012.

[14] Mir Md Jahangir Kabir, Shuxiang Xu, Byeong Ho Kang, and Zongyuan Zhao. A new multiple seeds based genetic algorithm for discovering a set of interesting Boolean association rules. *Expert Systems with Applications*, 74:55–69, 2017.

[15] Jesus Alcala-Fdez, Nicolo Flugy-Pape, Andrea Bonarini, and Francisco Herrera. Analysis of the effectiveness of the genetic algorithms based on extraction of association rules. *Fundamenta Informaticae*, 98(1):1–14, 2010.

[16] J. Mata, J. L. Alvarez, and J. C. Riquelme. An evolutionary algorithm to discover numeric association rules. *Proceedings of the ACM Symposium on Applied Computing*, pages 590–594, 2002.

[17] S E E Profile and S E E Profile. Mining Numeric Association Rules with Genetic Algorithms. *Artificial Neural Nets and Genetic Algorithms*, (February 2015), 2001.

[18] Christian Borgelt. Efficient implementations of apriori and eclat. *. . . Workshop on Frequent Itemset Mining Implementations*, (April), 2000.

[19] Mohammed J. Zaki. Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering*, 12(3):372–390, 2000.

[20] Min Wang, Qin Zou, and Caihui Liu. Multi-dimension association rule mining based on Adaptive Genetic Algorithm. *Proceedings of the International Conference on Uncertainty Reasoning and Knowledge Engineering, URKE 2011*, 2:150–153, 2011.

[21] Habiba Drias. Genetic algorithm versus memetic algorithm for association rules mining. *2014 6th World Congress on Nature and Biologically Inspired Computing, NaBIC 2014*, pages 208–213, 2014.

[22] Hong Guo and Ya Zhou. An algorithm for mining association rules based on improved genetic algorithm and its application. *3rd International Conference on Genetic and Evolutionary Computing, WGEC 2009*, pages 117–120, 2009.

[23] B. Minaei-Bidgoli, R. Barmaki, and M. Nasiri. Mining numerical association rules via multi-objective genetic algorithms. *Information Sciences*, 233:15–24, 2013.

[24] Mehmet Kaya and Reda Alhajj. Utilizing genetic algorithms to optimize membership functions for fuzzy weighted association rules mining. *Applied Intelligence*, 24(1):7–15, 2006.

[25] Y. Djenouri, H. Drias, Z. Habbas, and H. Mosteghanemi. Bees swarm optimization for web association rule mining. *Proceedings of the 2012 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology Workshops, WI-IAT 2012*, pages 142–146, 2012.

[26] Satya Ranjan Dash, Satchidananda Dehuri, and Susil Rayaguru. Discovering interesting rules from biological data using parallel genetic algorithm. *Proceedings of the 2013 3rd IEEE International Advance Computing Conference, IACC 2013*, pages 631–636, 2013.

[27] Youcef Djenouri, Ahcene Bendjoudi, Malika Mehdi, Nadia Nouali-Taboudjemat, and Zineb Habbas. Parallel association rules mining using GPUS and bees behaviors. *6th International Conference on Soft Computing and Pattern Recognition, SoCPaR 2014*, pages 401–405, 2014.

[28] Youcef Djenouri, Habiba Drias, and Amine Chemchem. A hybrid bees swarm optimization and Tabu search algorithm for association rule mining. *2013 World Congress on Nature and Biologically Inspired Computing, NaBIC 2013*, pages 120–125, 2013.

[29] R. J. Kuo and C. W. Shih. Association rule mining through the ant colony system for National Health Insurance Research Database in Taiwan. *Computers and Mathematics with Applications*, 54(11-12):1303–1318, 2007.

[30] Heikki Maaranen, Kaisa Miettinen, and Antti Penttinen. *On initial populations of a genetic algorithm for continuous optimization problems*, volume 37. 2007.

[31] Mir Md Jahangir Kabir, Shuxiang Xu, Byeong Ho Kang, and Zongyuan Zhao. Discovery of interesting association rules using genetic algorithm with adaptive mutation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9490:96–105, 2015.

[32] S. Kannimuthu and K. Premalatha. Discovery of high utility itemsets using genetic algorithm with ranked mutation. *Applied Artificial Intelligence*, 28(4):337–359, 2014.

[33] Ashish Ghosh and Bhabesh Nath. Multi-objective rule mining using genetic algorithms. *Information Sciences*, 163(1-3):123–133, 2004.

[34] S. Ventura. *Pattern Mining with Evolutionary Algorithms.* Berlin: Springer, 2016.

[35] Nisarg Pathak, Viral Shah, and Chandramohan Ajmeera. A memory efficient algorithm with enhance preprocessing technique for web usage mining. *Advances in Intelligent Systems and Computing*, 338:601–608, 2015.