



ISSN:1991-8178

## Australian Journal of Basic and Applied Sciences

Journal home page: www.ajbasweb.com



### Secure Multi Party Algorithm For Mining Of Association Rules

<sup>1</sup>T.Vijayalakshmi and <sup>2</sup>D.Kanthsamy<sup>1</sup>PG Scholar, Dept. of CSE Thiruvalluvar College of Engg. & Tech.<sup>2</sup>Assistant Professor, Dept. of CSE Thiruvalluvar College of Engg. & Tech.

#### ARTICLE INFO

##### Article history:

Article Received 12 January 2015

Revised 1 May 2015

Accepted 8 May 2015

##### Keywords:

Data Mining, Association Rules, ARM, Algorithms.

#### ABSTRACT

Association Rule Mining (AM) is one of the most popular data mining techniques. This paper provides a brief review about the existing Association Rules Data Mining. Association rule mining is normally performed in generation of frequent item sets and rule generation in which many researchers presented several efficient algorithms. Association rule mining has been extensively studied in the data mining community. The intention of this review is to present a broad classification of existing association rule mining algorithms and to motivate for the research.

© 2015 AENSI Publisher All rights reserved.

**To Cite This Article:** T.Vijayalakshmi and D.Kanthsamy., Secure Multi Party Algorithm For Mining Of Association Rules. *Aust. J. Basic & Appl. Sci.*, 9(21): 59-66, 2015

#### INTRODUCTION

Association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness. (Piatetsky-Shapiro, Gregory, 1991) Based on the concept of strong rules, Rakesh Agrawal *et al.* (1993) introduced association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets.

Following the original definition by Agrawal *et al.* (1993) the problem of association rule mining is defined as: Let be a set of  $n$  binary attributes called items. Let  $D = \{t_1, t_2, \dots, t_m\}$  be a set of transactions called the database. Each transaction in  $D$  has a unique transaction ID and contains a subset of the items in  $I$ . A rule is defined as an implication of the form where and. The sets of items  $X$  and  $Y$  are called antecedent (left-hand-side or LHS) and consequent (right-hand-side or RHS) of the rule respectively. Association rules are usually required to satisfy a user-specified minimum support and a user-specified minimum confidence at the same time. Association rule generation is usually split up into two separate steps:

1. First, minimum support is applied to find all frequent itemsets in a database.
2. Second, these frequent itemsets and the minimum confidence constraint are used to form rules.

While the second step is straightforward, the first step needs more attention. Many algorithms for generating association rules were presented over time. Some well known algorithms are Apriori, Eclat and FP-Growth, but they only do half the job, since they are algorithms for mining frequent item sets. Another step needs to be done after to generate rules from frequent item sets found in a database.

#### Data Mining:

Data mining is a detailed process of analyzing large amounts of data and picking out the relevant information. It refers to extracting or mining knowledge from large amounts of data (Han, J. and M. Kamber, 2000). It involves the following steps: cleaning and integrating data from data sources like databases, flat files, pre-treatment of selecting and transformation target data, mining the required knowledge and finally evaluation and presentation of knowledge. It is clearly explained pictorially in Figure 1.1. In data mining, association rule learning is a most popular methodology to identify the interesting relations between the data stored in large databases.

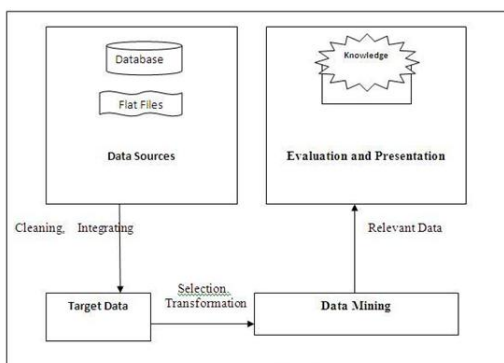


Fig. 1.1: Data Mining Process

**1.2 Association Rules:**

Association rule of data mining involves picking out the unknown inter-dependence of the data and finding out the rules between those items. Agrawal introduced association rules for Point Of Sale (POS) systems in supermarkets (Piatetsky-Shapiro, Gregory, 1991). A rule is defined as an implication of the form  $A \Rightarrow B$  where  $A \cap B \neq \emptyset$ . The left-hand side of the rule is called as antecedent. The right-hand side of the rule is called as consequent. Association rules are also used in many applications including Web usage Mining, Intrusion Detection and Bio-informatics (Das, A., et al., 2001).

**Support:**

Support is a fraction of transactions that contain an itemset. Frequencies of occurring patterns are indicated by support. The probability of a randomly

chosen transaction T that contain both itemsets X and Y is known as support. Mathematically it is represented as

$$P(X,Y) = \frac{\text{No. of transactions containing both X and Y}}{\text{Total no. of transactions}}$$

Equ... (1.1)

Equ... (1.1)

**Confidence:**

The confidence is defined as a conditional probability.

$$P(Y/X) = \frac{\text{No. of transactions containing both X and Y}}{\text{No. of transactions containing X}}$$

Equ...(1.2)

Table 1.1: Transactional Database

TID	Item Set
1	i1,i2
2	i1,i3,i5
3	i2,i3,im
.....	.....
m	i1,i3,im

**Application Areas:**

The various application areas in which association rules can be applied for extracting useful information from the huge dataset are:

- Market basket analysis
- Medical diagnosis
- Protein sequences
- Census data
- CRM of credit card business

**2. Problem Statement:**

If there existed a trusted third party, the players could surrender to him their inputs and he would perform the function evaluation and send to them the resulting output. In the absence of such a trusted third party, it is needed to devise a protocol that the

players can run on their own in order to arrive at the required output y.

**Objective:**

The main objective of this project is developing a protocol for the secure computation of the union of private subsets. The threshold and set inclusion protocol offers enhanced privacy and efficiency with respect to the UNIFI-KC(Unifying lists of locally Frequent Itemsets-Kantarcioglu and Clifton)

**Scope:**

This project aims to retrieve the files or data from all users to connect distributed shared data in which association rules can be applied for extracting useful information from the huge dataset in an efficient manner.

**System Analysis:****Existing System:**

Trusted third party is the one which communicates between distributed database and generates output for partial input given by the player (Han, J. and J. Pei, 2000). In the absence of trusted third party the distributed database communicates between themselves through a protocol (UNIFI-KC) of their own.

**Disadvantages of Existing System:**

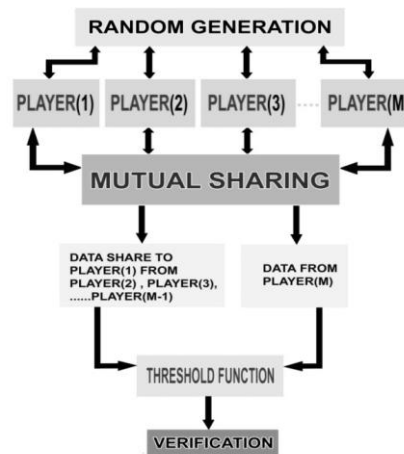
- Due to the absence of trusted third party

excess of information leakage takes place.

- Data loss occurs.
- Difficult to obtain accurate item set.
- Security issues arise.

**Proposed System:**

An alternative protocol for secure computation of the union of private subsets is proposed. This protocol improves upon UNIFI-KC (Han, J. and J. Pei, 2000) terms of simplicity and efficiency as well as privacy.



**Fig. 3.1:** System Architecture

The protocol that is proposed computes a parameterized family of functions, which is called threshold functions, in which the two extreme cases correspond to the problems of computing the union and intersection of private subsets. Those are in fact general-purpose protocols that can be used in other contexts as well. Another problem of secure multiparty computation that this method solved is the set inclusion problem (John, D., *et al.*, 2002).

Figure 3.1 illustrates that each player starts by creating random shares of his own input vector and selects random vectors that add up to in database. Now, all players send to all other players the corresponding shares in their input vector.

**Advantages of Proposed System:**

- Protocol does not depend on commutative encryption and oblivious transfer.
- The proposed method is faster.
- It improves privacy and efficiency.

**System Implementation:****Approach:**

The implementation of secure mining of association rules in horizontally distributed database is done using java eclipse tool.

**Implementation Phases:**

1. UNIFI-KC

2. Threshold
3. Set Inclusion

**Unifi-Kc:**

UNIFI-KC was suggested by Kantarcioglu and Clifton[9] for computing the unified list of all locally frequent item sets, without disclosing the sizes of the subsets nor their contents. The protocol is applied when the players already know the set of all item sets that are globally  $s$ -frequent, and they wish to proceed and compute. The input that each player  $P_m$  has at the beginning of Protocol UNIFI-KC is the collection  $C_s^{k,m}$  of the FDM algorithm. The output of the protocol is the union  $C_s^k = \bigcup_{m=1}^M C_s^{k,m}$ . In the first iteration of this computation  $k=1$ , and the players compute all  $s$ -frequent 1-items and the next iteration they compute all  $s$ -frequent 2-item sets, and so forth, until the  $k < L$  in which they find no  $s$ -frequent  $k$ -item sets.

After computing that union, the players proceed to extract from  $C_s^k$  the subset  $F_s^k$  that consists of all  $k$ -item sets that are globally  $s$ -frequent is done using the protocol UNIFI-KC. Finally, by applying the procedure from  $k=1$  until the first value of  $k < L$  for which the resulting set  $F_s^k$  is empty, the players may recover the full set of all globally  $s$ -frequent item sets.

Protocol UNIFI-KC works as follows: First, each player adds to his private subset with fake item

sets, in order to hide its size. Then, the players jointly compute the encryption of their private subsets by applying on those subsets a commutative encryption, where each player adds in his turn, his own layer of encryption using his private secret key. At the end of that stage, every item set in each subset is encrypted by all of the players and the usage of a commutative encryption scheme ensures that all item sets are,

eventually, encrypted in the same manner. Then, they compute the union of those subsets in their encrypted form. Finally, they decrypt the union set and remove from it item sets which are identified as fake and broadcasts the results.

#### Flowchart:

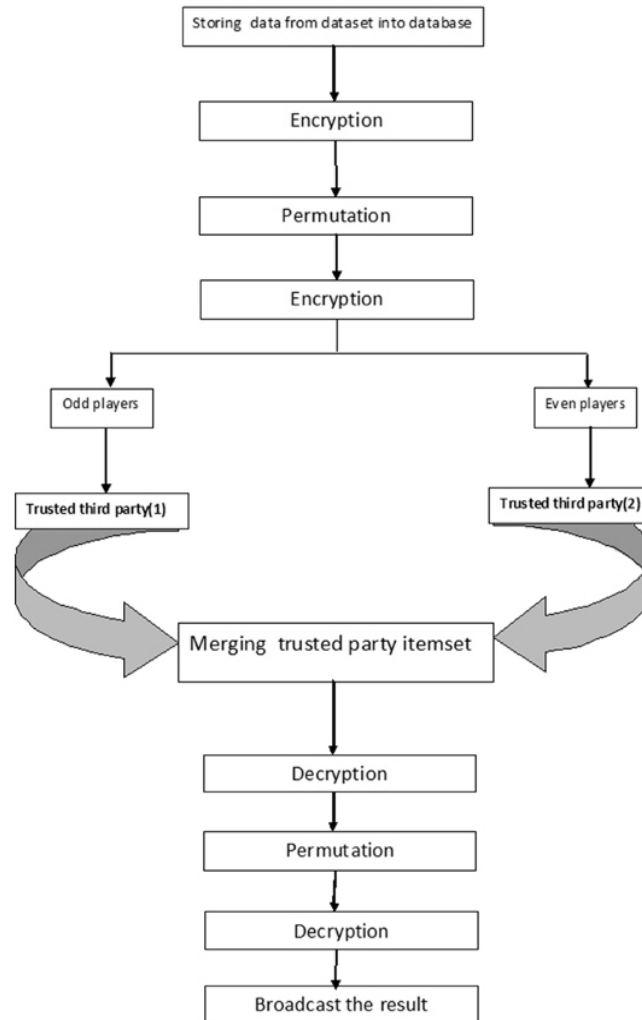


Fig. 4.1: Process Of Unifi-Kc

#### Algorithm:

**Input:** Each player  $P_m$  has an input set  $C_s^{k,m} \subseteq Ap(F_s^{k-1})$ ,  $1 \leq m \leq M$ .

**Output:**  $C_s^k = \bigcup_{m=1}^M C_s^{k,m}$ .

1: **Phase 0: Getting started**

2: The players decide on a commutative cipher and each player  $P_m$ ,  $1 \leq m \leq M$ , selects a random secret encryption key  $K_m$ .

3: The players select a hash function  $h$  and compute  $h(x)$  for all  $x \in Ap(F_s^{k-1})$ .

4: Build a lookup table  $T = \{(x, h(x)) : x \in Ap(F_s^{k-1})\}$ .

5: **Phase 1: Encryption of all itemsets**

6: **for all** Player  $P_m$ ,  $1 \leq m \leq M$ , **do**

7: Set  $X_m = \emptyset$ .

8: **for all**  $x \in C_s^{k,m}$  **do**

9: Player  $P_m$  computes  $E_{K_m}(h(x))$  and adds it to  $X_m$ .

10: **end for**

- 11: Player  $P_m$  adds to  $X_m$  faked itemsets until its size becomes  $|Ap(F_s^{k-1})|$ .
- 12: **end for**
- 13: **for**  $i = 2$  to  $M$  **do**
- 14:   **for** all  $1 \leq m \leq M$  **do**
- 15:      $P_m$  sends a permutation of  $X_m$  to  $P_{m+1}$ .
- 16:      $P_m$  receives from  $P_{m-1}$  the permuted  $X_{m-1}$ .
- 17:      $P_m$  computes a new  $X_m$  as the encryption of the permuted  $X_{m-1}$  using the key  $K_m$ .
- 18:   **end for**
- 19: **end for**
- 20: **Phase 2: Merging itemsets**
- 21: Each odd player sends his encrypted set to player  $P_1$ .
- 22: Each even player sends his encrypted set to player  $P_2$ .
- 23:  $P_1$  unifies all sets that were sent by the odd players and removes duplicates.
- 24:  $P_2$  unifies all sets that were sent by the even players and removes duplicates.
- 25:  $P_2$  sends his permuted list of itemsets to  $P_1$ .
- 26:  $P_1$  unifies his list of itemsets and the list received from  $P_2$  and then removes duplicates from the unified list. Denote the final list by  $EC_s^k$ .
- 27: **Phase 3: Decryption**
- 28: **for**  $m = 1$  to  $M - 1$  **do**
- 29:    $P_m$  decrypts all itemsets in  $EC_s^k$  using  $K_m$ .
- 30:    $P_m$  sends the permuted (and  $K_m$ -decrypted)  $EC_s^k$  to  $P_{m+1}$ .
- 31: **end for**
- 32:  $P_M$  decrypts all itemsets in  $EC_s^k$  using  $K_M$ ; denote the resulting set by  $C_s^k$ .
- 33:  $P_M$  uses the lookup table  $T$  to replace hashed values with the actual itemsets, and to identify and remove faked itemsets.
- 34:  $P_M$  broadcasts  $C_s^k$ .

### Threshold:

In this protocol (Yuh-Jiuan Tsay, Jiunn-Yann Chiang, 2005; Agrawal, R., *et al.*, 1993) each player starts by creating random shares of his input vector,  $P_m$  selects  $M$  random vectors in database that add up to  $\mathbf{b}_m$ ,  $1 \leq m \leq M$ . Then all players send to all other players the corresponding shares in their input vector. Now, player  $P_l$ ,  $1 \leq l \leq M$ , adds the shares that he got and arrives at his share,  $s_l$ , in the sum vector 'a'. Furthermore, any  $M - 1$  vectors out of  $M$  do not reveal any information on the sum 'a'. Then

all players, apart from the last one, send their shares to  $P_1$  who adds them up to get the share  $s$ . Now, players  $P_1$  and  $P_M$  hold additive shares of the sum vector  $\mathbf{a}$ :  $P_1$  has  $s$ ,  $P_M$  has  $s_M$ , and  $\mathbf{a} = (s + s_M)$  mod  $(M+1)$ . It is now needed to check for each component  $1 \leq i \leq n$  whether

$$(s(i) + s_M(i)) \bmod (M+1) < t \quad \text{Equ...} \quad (6.1)$$

Whenever inequality (4.1) holds, we set  $b(i)=0$  otherwise, we set  $b(i)=1$

### Algorithm:

- Input:** Each player  $P_m$  has an input binary vector  $\mathbf{b}_m \in \mathbb{Z}_2^n$ ,  $1 \leq m \leq M$ .
- Output:**  $\mathbf{b} := T_t(\mathbf{b}_1, \dots, \mathbf{b}_M)$ .
- 1: Each  $P_m$  selects  $M$  random share vectors  $\mathbf{b}_{m,\ell} \in \mathbb{Z}_{M+1}^n$ ,  $1 \leq \ell \leq M$ , such that  $\sum_{\ell=1}^M \mathbf{b}_{m,\ell} = \mathbf{b}_m \bmod (M+1)$ .
  - 2: Each  $P_m$  sends  $\mathbf{b}_{m,\ell}$  to  $P_\ell$  for all  $1 \leq \ell \neq m \leq M$ .
  - 3: Each  $P_\ell$  computes  $\mathbf{s}_\ell = (s_\ell(1), \dots, s_\ell(n)) := \sum_{m=1}^M \mathbf{b}_{m,\ell} \bmod (M+1)$ .
  - 4: Players  $P_\ell$ ,  $2 \leq \ell \leq M - 1$ , send  $\mathbf{s}_\ell$  to  $P_1$ .
  - 5:  $P_1$  computes  $\mathbf{s} = (s(1), \dots, s(n)) := \sum_{\ell=1}^{M-1} \mathbf{s}_\ell \bmod (M+1)$ .
  - 6: **for**  $i = 1, \dots, n$  **do**
  - 7:   If  $(s(i) + s_M(i)) \bmod (M+1) < t$  set  $b(i) = 0$  otherwise set  $b(i) = 1$ .
  - 8: **end for**
  - 9: Output  $\mathbf{b} = (b(1), \dots, b(n))$ .

### Flowchart:

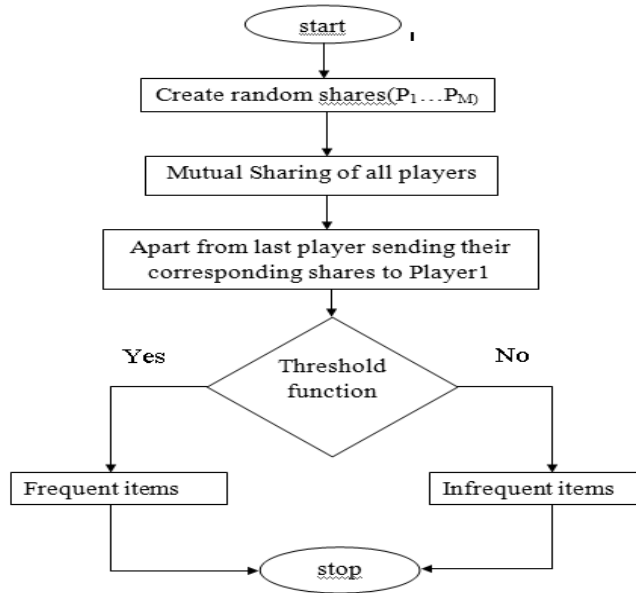


Fig. 4.2: Process Of Threshold Computation

**Set Inclusion(SETINC):**

Protocol SETINC involves three players:  $P_1$  has a vector  $s=(s(1), \dots, s(n))$  of elements in some ground set  $\Omega$  and  $P_M$ , on the other hand, has a vector  $\Theta=(\Theta(1), \dots, \Theta(n))$  of subsets of that ground set. The required output is a vector  $b=(b(1), \dots, b(n))$  that describes the corresponding set inclusions in the following manner:  $b(i) = 0$  if  $s(i) \in \Theta(i)$  and  $b(i) = 1$  if  $s(i) \notin \Theta(i)$ ,  $1 \leq i \leq n$ .

The protocol starts with players  $P_1$  and  $P_M$  agreeing on a keyed hash function  $h_K(\cdot)$  and a

corresponding secret key  $K$ . Consequently,  $P_1$  converts his sequence of elements  $s = (s(1), \dots, s(n))$  into a sequence of corresponding "signatures"  $s' = (s'(1), \dots, s'(n))$ , where  $s'(i) = h_K(i, s(i))$  and  $P_M$  does a similar conversions to the subsets that he holds. Then  $P_1$  sends  $s'$  to  $P_2$ , and  $P_M$  sends to  $P_2$  the subsets  $\Theta'(i)$ ,  $1 \leq i \leq n$ , where the elements within each subset are randomly permuted. Finally  $P_2$  performs the relevant inclusion verifications (Agrawal, R. and R. Srikant, 1994) on the signature values.

**Flowchart:**

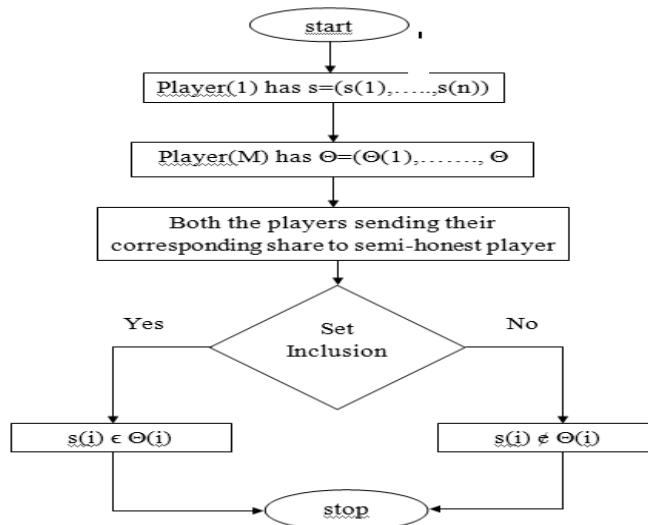


Fig. 4.3 Process Of Set Inclusion:

**Algorithm:**

- Input:**  $P_1$  has a vector  $\mathbf{s} = (s(1), \dots, s(n))$  and  $P_M$  has a vector  $\Theta = (\Theta(1), \dots, \Theta(n))$ , where for all  $1 \leq i \leq n$ ,  $s(i) \in \Omega$  and  $\Theta(i) \subseteq \Omega$  for some ground set  $\Omega$ .
- Output:** The vector  $\mathbf{b} = (b(1), \dots, b(n))$  where  $b(i) = 0$  if  $s(i) \in \Theta(i)$  and  $b(i) = 1$  otherwise,  $1 \leq i \leq n$ .
- 1:  $P_1$  and  $P_M$  agree on a keyed-hash function  $h_K(\cdot)$  and on a secret key  $K$ .
  - 2:  $P_1$  computes  $\mathbf{s}' = (s'(1), \dots, s'(n))$ , where  $s'(i) = h_K(i, s(i))$ ,  $1 \leq i \leq n$ .
  - 3:  $P_M$  computes  $\Theta' = (\Theta'(1), \dots, \Theta'(n))$ , where  $\Theta'(i) = \{h_K(i, \theta) : \theta \in \Theta(i)\}$ ,  $1 \leq i \leq n$ .
  - 4:  $P_1$  sends to  $P_2$  the vector  $\mathbf{s}'$ .
  - 5:  $P_M$  sends to  $P_2$  the vector  $\Theta'$  in which each  $\Theta(i)$  is randomly permuted.
  - 6: For all  $1 \leq i \leq n$ ,  $P_2$  sets  $b(i) = 0$  if  $s'(i) \in \Theta'(i)$  and  $b(i) = 1$  otherwise.
  - 7:  $P_2$  broadcasts the vector  $\mathbf{b} = (b(1), \dots, b(n))$ .

**Conclusion and Future Work:****Conclusion:**

The main ingredients in proposed system is a novel secure multi-party protocol for computing the union (or intersection) of private subsets that each of the interacting players hold and another ingredient is a protocol that tests the inclusion of an element held by one player in a subset held by another. The proposed multi party protocols provides efficient mining of association rules which is proved through performance evaluation of the system.

**Future Work:**

The future work is aimed at extending the approach to enhance the protocol for inequality verifications. This might enable to further improve the communication and computational costs of the proposed protocol. The other area of focus is to apply the proposed protocol implementation in distributed association rule mining in the vertical setting

**REFERENCES**

- Piatetsky-Shapiro, Gregory, 1991. Discovery, analysis, and presentation of strong rules, in Piatetsky-Shapiro, Gregory; and Frawley, William J.; eds., Knowledge Discovery in Databases, AAAI/MIT Press, Cambridge, MA.
- Agrawal, R., T. Imieliński, A. Swami, 1993. "Proceedings of the 1993 ACM SIGMOD international conference on Management of data - SIGMOD '93". p: 207. doi:10.1145/170035.170072.
- Agrawal, R., T. Imieliński and A.N. Swami, 1993. Mining association rules between sets of items in large databases. In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, P. Buneman and S. Jajodia, Eds. Washington, D.C., pp: 207-216.
- Agrawal, R. and R. Srikant, 1994. Fast algorithms for mining association rules. In Proc. 20th Int. Conf. Very Large Data Bases, VLDB, J. B. Bocca, M. Jarke, and C. Zaniolo, Eds. Morgan Kaufmann, pp: 487-499.
- Srikant, R. and R. Agrawal, 1996. Mining quantitative association rules in large relational tables. In Proceedings of the 1996 ACM SIGMOD international conference on Management of data. ACM Press, pp: 1-12.
- Cheung, D.W.-L., S.D. Lee and B. Kao, 1997. A general incremental technique for maintaining discovered association rules. In Database Systems for Advanced Applications, pp: 185-194.
- Han, J. and M. Kamber, 2000. Data Mining Concepts and Techniques. Morgan Kaufmann.
- Pei, J. and J. Han, 2000. Can we push more constraints into frequent pattern mining? In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM Press, pp: 350-354.
- Han, J. and J. Pei, 2000. Mining frequent patterns by pattern-growth: methodology and implications. ACM SIGKDD Explorations Newsletter, 2(2): 14-20.
- Das, A., W.-K. Ng and Y.-K. Woon, 2001. Rapid association rule mining. In Proceedings of the tenth international conference on Information and knowledge management. ACM Press, pp: 474-481.
- John, D., Holt, Soon M. Chung, 2002. "Mining association rules using inverted hashing and pruning", Information Processing Letters, 83(4): 211-220.
- Yi-Chung Hu, Ruey-Shun Chen, Gwo-Hsiung Tzeng, 2003. "Discovering fuzzy association rules using fuzzy partition methods", Knowledge-Based Systems, 16(3): 137-147.
- Feng-Hsu Wang, Hsiu-Mei Shao, 2004. "Effective personalized recommendation based on time-framed navigation clustering and association mining", Expert Systems with Applications, 27(3): 365-37.
- Yuh-Jiuan Tsay, Jiunn-Yann Chiang, 2005. "BAR: an efficient method for mining association rules", Knowledge-Based Systems, 18(2-3): 99-10.
- Guoqing Chen, Hongyan Liu, Lan Yu, Qiang Wei, Xing Zhang, 2006. "A new approach to classification based on association rule mining", Decision Support Systems, 42(2): 674-68.

Frans Coenen, Paul Leng, 2007. "The effect of threshold values on association rule based classification accuracy", Data & Knowledge Engineering, 60(2): 345-360.

He Jiang; Yuanyuan Zhao; Xiangjun Dong, 2008."Mining Positive and Negative Weighted

Association Rules from Frequent Itemsets Based on Interest," Computational Intelligence and Design,ISCID '08. International Symposium on, 2: 242,245, 17-18.