ISSN: 0975-766X
CODEN: IJPTFI
Available Online through ***Research Article***
www.ijptonline.com

ON IMPLEMENTINGITEMSET MINING USING CONCEPT LATTICE

A. Manaswi D. Rajani, M.Sai Harika
School of Information Technology and Engineering, VIT University, Vellore, India.
Email: manaswi.arja@gmail.com

Received on 25-10-2016 *Accepted on 02-11-2016*

Abstract:

Itemsets mining involves finding the frequent itemsets from a given transaction database. Several techniques are available in the literature to mine the frequent itemsets. Formal Concept Analysis (FCA) is a mathematical framework for knowledge representation and data analysis. The notion of formal concept in FCA generalizes the itemsets by considering the itemsets as intent and the support as the cardinality of the extent. In this paper we systematically analyze the FCA based itemsets mining. We demonstrate the process with experiments on standard datasets.

Keywords: Association rules, Concept lattice, Formal concept analysis, Frequent items.

1. Introduction

Association rule mining is the most investigating approach in order to extract frequent patterns or associations from the item set[3]. Originally applied in super market settings, itemsets mining involves finding frequent itemsets[2] thatis, set of items that are frequently bought together in transaction of customers. Applications of association rules are in telecommunication networks, market and risk management, inventory control etc[15]. A very good survey on association rule mining was made by Ceglar and Roddick[9].

Recently, there are growing interests in itemsets mining using Formal Concept Analysis (FCA), which is a branch of applied mathematics. FCA is originated in 1980’s by Rudolf Wille as a mathematical theory (Wille,1982) [24]. It is a form of knowledge representation which represents the data as a formal context. FCA formalizes the concepts and can be utilized in various areas such as software engineering, knowledge discoveryand information retrieval [14] [26] [27] [28][29] [34].For knowledge processing tasks fuzzy formal concepts and their concept lattice structure were introduced by several researchers [30] [31] [32] [33] [35].

With its mathematical properties, FCA is widely applied for the task of itemsets mining [23] [25]. This paper is aimed at providing a systematic understanding on application of FCA for itemsets mining. The rest of the paper is organized as follows. Section 2 provides a background on association rule mining and FCA. Section 3 deals with rule mining using FCA and the experimental results are discussed in Section 4. Section 5 includes a brief discussion on the rule mining using FCA.

2. Background

Association Rule Mining (ARM) is one of the core tasks of data mining. It was popularized due to R.Agrawal's article published in 1993 [1]. Implication of the form $X \rightarrow Y$ is an association rule. The expression describes the relationship between itemsets X and Y, where $X, Y \subset I$ (I be set of items) and $X \cap Y = \phi$. Consider a transaction T in a database D, the rule $X \rightarrow Y$ demonstrate that whenever a transaction T is to made which contains X then T will also contains Y[23].

Each association rule has two quality measurements namely support and confidence. Support denotes the percentage of transactions that contain both X and Y. Confidence is the percentage of transactions containing X that also contain Y [15]. The rule is interesting if its support and confidence are equal to or greater than user specified minimum support and minimum confidence thresholds, respectively. 100% confidence with association rule is called exact association rules (implications) and those with less than 100% confidence are called approximate association rules [19]. For a detailed description and illustration of ARM, interested readers can refer [2][9][15].

2.1. Formal Concept Analysis

Formal concept analysis (FCA) is a mathematical theory of concepts which investigates and represents the knowledge in the form of two key components such as formal concepts and their ordered hierarchical structure known as concept lattice [21].

A formal context is a triple $k = (G, M, I)$ where G is a set of objects, M a set of attributes (or items), a binary relation $I \subseteq G \times M$ [13]. Table 1 shows an example of a formal context. Two operators ↑ and ↓ are defined for every $X \in G$ and $Y \in M$ where

$$X^{\uparrow} = \{m \in M \mid \text{for each } g \in G\text{: } (g, m) \in I\} \quad (1)$$

$$Y^{\downarrow} = \{g \in G \mid \text{for each } m \in M\text{: } (g, m) \in I\} \quad (2)$$

The two operator's ↑ and ↓ are called Concept forming operators [24]. The set (X, Y) is called a Formal concept(C). X is called Extension of the concept (Ext(C)), and Y is called Intension of the concept (Int(C)) [19]. In the standard formal context, the constraint for a formal concept is that all objects in the extent must have all properties in the intent and all properties in the intent must be possessed by all objects in the extent [14].

A Galois connection[14] is defined between the sets G and M, as a pair (f, g) where $f : 2^G \rightarrow 2^M$ and $g : 2^M \rightarrow 2^G$ satisfying the following conditions for $a, a_1, a_2 \in G$ and $b, b_1, b_2 \in M$

$$a_1 \subseteq a_2 \Rightarrow f(a_2) \subseteq f(a_1). \quad (3)$$

$$b_1 \subseteq b_2 \Rightarrow g(b_2) \subseteq g(b_1). \quad (4)$$

$$a \subseteq g(b) \Rightarrow b = f(a). \quad (5)$$

Table 1: Formal Context.

	a	**b**	**c**	**d**	**e**
1	X		X	X	
2		X	X		X
3	X	X	X		X
4		X			X

Every Galois connection forms a concept-forming operator for a particular formal context [20]. The concepts can be arranged in a hierarchically ordered structure known as concept lattice [18]. Moving towards apex in the lattice structure shows the generalization relationship among the concepts and moving towards base shows the specialization relationship among the concepts. The subconcept-superconcept relation represents the partial order relation between the formal concepts [18]. For more details on FCA and its applications, readers can refer [8][18][20].

3. Rule mining using FCA

By using formal concepts one can generate the association rules. The closed frequent itemsets are considered as extent of the concept. The support of the itemsets depicts the length of the corresponding intent. The algorithms available in the literature to generate the formal concepts from the given context include Bordat, Ganter (Next Closure), Chein, Norris, Godin, Nourine, Carpineto, Object Intersections, Next Neighbors and Valtchev, etc. The analysis of these algorithms can be found in [6][16][20].In this paper, we generate the concepts using "Object Intersections" algorithm [8]. The main idea behind this algorithm is that every concept extent is the intersection of attribute extents and every concept's intent is the intersection of object intents. Here the concepts are generated incrementally. The algorithm is given in Table2.

Concepts can be generated by calculating all intersections of object intents and then by determining the extent of each such concept. In order to form all possible intersections, one object is considered at a time and the intersection between its intent and each concept's intent in the current set of concepts. The current set of concepts contains concepts generated by objects that have already been examined. The dual version of this algorithm can be used to generate the set of concepts i.e. by taking each attribute at a time. The choice to choose one among the dual versions of the algorithm is to be done by the consideration of how the number of objects and the number of attributes affects the algorithm efficiency.The worst case complexity of this algorithm is $O(|G||C||M|)$ where, $|G|$ is the number of objects, $|C|$ is the number of concepts and $|M|$ is the number of attributes.

Table 2: Object Intersections algorithm.

Algorithm(Object Intersections): To generate the set of concepts

Input: Context (O, A, R)

Output: The set C of all concepts of (O, A, R)

Method:

1. $C = \{(A^{\downarrow}, A)\}$;
2. for each $o \in O$
3.   for each $(X,Y) \in C$
4. $i = Y \cap \{o^{\uparrow}\}$;
5.   if i is different from any concept intent in C then
6. $C = C \cup \{(i^{\downarrow}, i)\}$

Algorithm Description:

Initially, attribute set A is taken and its extent $(A^{\downarrow})$ is determined. The $(A^{\downarrow}, A)$ is included in the list of concepts. For every object o in object set O, consider each concept(X, Y) from the list of concepts generated.

- Find the intersection of the intent with the intent of o and store in the variable i.

- If *i* does not match with any of the intents of concepts generated previously,
- Compute the extent of *i* i.e. $i^{\downarrow}$.
- Include ($i^{\downarrow}$, *i*) in the concepts list.

Example. For the context in Table 1, concepts are generated as follows.

- *O* : {1 2 3 4} ; *A* : { a b c d e }
- $A^{\downarrow} = \varphi$; *C* = (φ, a b c d e)
- *o* = 1
  - *C* = (φ, a b c d e)
    - → $i = (a\ b\ c\ d\ e) \cap (1)^{\uparrow} = (a\ b\ c\ d\ e) \cap (a\ c\ d) = (a\ c\ d)$
    - → $(i)^{\downarrow} = (a\ c\ d)^{\downarrow} = 1$
    - → *C* = { (φ, a b c d e) (1, a c d) }
    - → (1, a c d) is added to the list of concepts *C*
- *o* = 2
  - *C*=(φ, a b c d e)
    - → $i = (a\ b\ c\ d\ e) \cap (2)^{\uparrow} = (a\ b\ c\ d\ e) \cap (b\ c\ e) = (b\ c\ e)$
    - → $(i)^{\downarrow} = (b\ c\ e)^{\downarrow} = 2\ 3$
    - → (2 3,b c e) is added to the list of concepts *C*
  - *C* = (1, a c d)
    - → $i = (a\ c\ d) \cap (2)^{\uparrow} = (a\ c\ d) \cap (b\ c\ e) = (c)$
    - → $(i)^{\downarrow} = (c)^{\downarrow} = 1\ 2\ 3$
    - → (1 2 3,c) is added to the list of concepts *C*
- *o* = 3
  - *C* = (φ, a b c d e)
    - → $i = (a\ b\ c\ d\ e) \cap (3)^{\uparrow} = (a\ b\ c\ d\ e) \cap (a\ b\ c\ e) = (a\ b\ c\ e)$
    - → $(i)^{\downarrow} = (a\ b\ c\ e)^{\downarrow} = 3$
    - → (3,a b c e) is added to the list of concepts *C*

- C = (1, a c d)
  - → i = (a c d) ∩ $(3)^{\uparrow}$ = (a c d) ∩ (a b c e) = (a c)
  - → $(i)^{\downarrow}$ = $(\text{a c})^{\downarrow}$=1 3
  - → (1 3,a c) is added to the list of concepts C

- C = (2 3,b c e)
  - → i = (b c e) ∩ $(3)^{\uparrow}$ = (b c e) ∩ (a b c e) = (b c e)
  - → $(i)^{\downarrow}$ = $(\text{b c e})^{\downarrow}$= 2 3
  - → (2 3, b c e) is already generated. It is not added to the list of concepts C.

- o = 4
  - C = (φ, a b c d e)
    - → i = (a b c d e) ∩ $(4)^{\uparrow}$ = (a b c d e) ∩ (b e) = (b e)
    - → $(i)^{\downarrow}$ = $(\text{b e})^{\downarrow}$= 2 3 4
    - → (2 3 4,b e) is added to the list of concepts C
  - C = (1, a c d)
    - → i = (a c d) ∩ $(4)^{\uparrow}$ = (a c d) ∩ (b e) = φ
  - C = (2 3,b c e)
    - → i = (b c e) ∩ $(4)^{\uparrow}$ = (b c e) ∩ (b e) = (b e)
    - → $(i)^{\downarrow}$ = $(\text{b e})^{\downarrow}$= 2 3 4
    - → (2 3 4, b e) is already generated. It is not added to the list of concepts C.
  - C = (1 2 3,c)
    - → i = (c) ∩ $(4)^{\uparrow}$ = (c) ∩ (b e) = φ
  - C = (1 3,a c)
    - → i = (a c) ∩ $(4)^{\uparrow}$ = (a c) ∩ (b e) = φ
  - C = (3, a b c e)
    - → i = (a b c e) ∩ $(4)^{\uparrow}$ = (a b c e) ∩ (b e) = b e

→ $(i)^{\downarrow} = (\text{b e})^{\downarrow} = 2\ 3\ 4$

→ (2 3 4, b e) is already generated. It is not added to the list of concepts *C*.

The list of concepts generated for the context in Table 1 is shown in Table 3.

Table 3: Formal concepts.

S.No.	Concepts
1	(φ, a b c d e)
2	(1, a c d)
3	(2 3,b c e)
4	(1 2 3,c)
5	(3,a b c e)
6	(1 3,a c)
7	(2 3 4,b e)
8	(1 2 3 4, φ)

Concept lattice structure provides the hierarchical visualization of the concepts. The lattice can be built by first creating all set of concepts and then setting the edges between them. The edges between the concepts can be set using the Concepts Cover algorithm [8]. Table 4shows the algorithm. The complexity of this algorithm is $O(|C||M|(|G|+|M|))$ where $|G|$ denotes the number of objects, $|C|$ denotes the number of concepts and $|M|$ denotes the number of attributes.

Table 4: **Concepts Cover algorithm.**

Algorithm(Concepts Cover): To generate concept lattice

Input: Context(*O*, *A*, *R*)

Output: Concept lattice L=(*C*,*E*) of (*O*, *A*, *R*)

Method:

1. Find *C* with the Object Intersections algorithm
2. Edges(*C*(*O*, *A*, *R*))

function Edges(*C*, (*O*, *A*, *R*))

```
1. for each (X,Y) ∈C
2.     set count of any concept in C to 0
3.     for each a∈A\Y
4.         i = X ∩ {a↓}
5.         Find (X1,Y1) ∈C such that X1= i
6.         count (X1,Y1) = count (X1,Y1) + 1
7.         if ( |Y1| - |Y| )= count(X1,Y1) then
8.             Add edge (X1,Y1) → (X,Y) to E
```

Algorithm Description:

Each concept (X, Y) is examined at a time and its lower neighbors are found. The candidate neighbor (X_1, Y_1) of a concept (X, Y) is generated by considering for each attribute m not contained in Y. Here, X_1 is computed by taking the intersection of X and the attribute extent of a. Y_1 is computed from the concepts generated previously using X_1 as the key. For selecting maximal candidates, a counter is maintained for each concept. At the outset, the counter is initialized to zero. The counter of a concept (X_1, Y_1) increases whenever that concept is retrieved as a candidate. If the difference between cardinalities of Y_1 and Y becomes equal to the counter of (X_1, Y_1), then the concept (X_1, Y_1) is considered as a child of (X, Y). An edge is drawn from (X_1, Y_1) to (X, Y). From these steps, it can be assured that only maximal candidates are selected and each maximal candidate is taken exactly once.

Example.

- O:{1 2 3 4 } A:{a b c d e}
- Consider the concept (1 2 3 4, φ)
- To find the children of this concept, initially set the count value to zero for all the concepts.
- a=a
    - i = (1 2 3 4) ∩ $a^{\downarrow}$ = (1 2 3 4) ∩ (1 3) = (1 3)
    - (X_1,Y_1) = (1 3,a c) ; count(1 3,a c)=1
    - The condition in step 7 of the algorithm fails.
- a=b

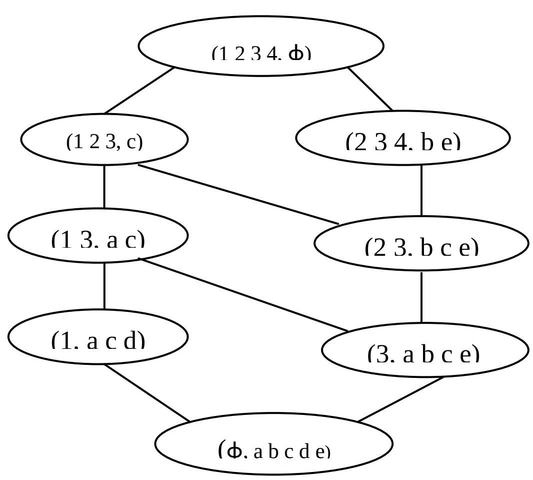
  - i =(1 2 3 4) ∩ $b^{\downarrow}$ = (1 2 3 4) ∩ (2 3 4) = (2 3 4)
  - (X_1,Y_1) = (2 3 4,b e) ; count(2 3 4,b e)=1
  - The condition in step 7 of the algorithm fails.

- a=c
  - i =(1 2 3 4) ∩ $c^{\downarrow}$ = (1 2 3 4) ∩ (1 2 3) = (1 2 3)
  - (X_1,Y_1) = (1 2 3,c) ; count(1 2 3,c)=1
  - The condition in step 7 of the algorithm satisfies.
  - Hence, add the edge (1 2 3, c) → (1 2 3 4, φ).

- a=d
  - i =(1 2 3 4) ∩ $d^{\downarrow}$ = (1 2 3 4) ∩ (1) = (1)
  - (X_1,Y_1) = (1, a c d) ; count(1, a c d)=1
  - The condition in step 7 of the algorithm fails.

- a=e
  - i =(1 2 3 4) ∩ $e^{\downarrow}$ = (1 2 3 4) ∩ (2 3 4) = (2 3 4)
  - (X_1,Y_1) = (2 3 4,b e) ; count(2 3 4,b e)=2
  - The condition in step 7 of the algorithm satisfies.
  - Hence, add the edge (2 3 4, b e)→ (1 2 3 4, φ).

Figure 1. Concept lattice



Association Rules: From the concept lattice, association rules are generated. The rules with 100% confidence are called implications. "Find Implications" algorithm is used to mine the implications [8]. Concept lattice of the context is used as input to the algorithm. An implication $P \rightarrow Q$ holds in a context (G, M, I) if and only if $(Q^{\downarrow}, Q^{\downarrow\uparrow}) \geq (P^{\downarrow}, P^{\downarrow\uparrow})$. In other terms, $P \rightarrow Q$ holds if and only if the largest concept containing P as a part of its intent is also described by Q.

The implications can be generated from a concept (X, Y) in the form $P \rightarrow Q$, with $P \cap Q = \Phi$, $P \subseteq Y$, $Q = Y \setminus P$, provided that there is no parent (W, Z) of (X, Y) such that $P \subseteq Z$. If the last condition is satisfied, then P is consistent with the parents of (X, Y).The algorithm is given in Table 5. *lset*updating needs time for each concept's parent at most proportional to $k^2|M|$, where k is the largest size of *lset* and $|M|$ is the number of attributes. Thus, Find Implications algorithm complexity is at most proportional to $O(|C|k^2|M|q)$, where $|C|$ is the number of concepts and q is the largest number of parents per concept.

Table 5: Find Implications Algorithm.

Algorithm(Find Implications): To generate exact association rules

Input: A concept lattice L of context (O, A, R)

Output: Implications that hold in (O, A, R)

1. $fimpls = \varphi$
2. for each$(X,Y) \in L$
3. $fimpls = fimpls \cup$ implsFromConcept$((X,Y),L)$

function implsFromConcept$((X,Y),L)$

/*Returns the implications generated from concept (X,Y)*/

1. $lset = Y$

   /*Contains the antecedent of the implications derivable from concept*/
2. for each(W,Z) parent of (X,Y)
3. $ulset = \varphi$
4. for each $l \in lset$
5. if $l \in Z$ then
6. $ulset = ulset \cup \{l\}$

```
        else
7.          cset =φ
8.          for each a∈Y\ l
9.   cset = cset U { l ∪{a}}
10.         for each c∈cset
11.             if check(c, l, Z, lset)=true then
12. ulset = ulset∪ {c}
13. lset=ulset
14. cimpls = φ
15. for each l ∈lset
16. cimpls = cimpls∪ {l→Y\l}
17.  return cimpls
function check( c, l, Z, lset)
1.   if c∉Z and there is no element ⊆c in lset \ {l} then
2.     return true
```

Algorithm Description:

From all the concepts generated, each concept is taken at time and the procedure impls From Concept is called to derive all the implications from that concept.

In the procedure implsFromConcept, the intent of the concept is taken as *lset*. One parent at a time is examined by the algorithm, maintaining and updating a set *lset* containing all the most general elements that are consistent with the parents. The *lset* is updated in the following way. Initially, it contains the most general elements of *Y*, i.e., those formed by single attributes. Then, for each parent of (*X*, Y), the procedure deletes all the elements that are inconsistent with the parent from *lset*, and replaces each with its most general consistent specializations (a specialization of an element *l* in *lset* is any element of Y that is more specific than *l*). Finally, it removes the consistent specializations that are more specific than (or equal to) some other elements of *lset*. For each element in the updated *lset*, the rule $l \rightarrow Y \setminus l$ is formed.

Example:

Consider the concept (2 3, b c e). From the concept lattice in Fig.1, the parents of this concept are (1 2 3, c) and (2 3 4, b e).

Parent (*W*, *Z*) = (1 2 3, c)

- *lset* ={b, c, e}
  - *l* = b
    - → The condition in step 5 of the algorithm satisfies.
    - → *ulset* = b
  - *l* = c
    - → The condition in step 5 of the algorithm fails.
    - → *cset* = (c b),(c e)
    - → Both (c b), (c e) are inconsistent. i.e., fails the condition in check function.
  - *l* = e
    - → The condition in step 5 of the algorithm satisfies.
    - → *ulset* = e
- *lset* = {b, e}

Parent (*W*, Z) = (2 3 4, b e)

- *lset* = {b, e}
  - *l* = b
    - → The condition in step 5 of the algorithm fails.
    - → *cset* = (b c),(b e)
    - → (b c) satisfies the condition in check function.
    - → *ulset* = (b c)
  - *l* = e
    - → The condition in step 5 of the algorithm fails.
    - → *cset* = (e b),(e c)

→ (e c) satisfies the condition in check function.

→ *ulset* = (e c)

- *lset*={(b c),(e c)}

The implications that are derived from the concept (2 3, b c e) are b c → e and e c → b. Implications generated for the context in Table 1 are given in Table 6.

Table 6: Implications generated using FCA.

No.	Implications
1	b d → a c e
2	e d → a b c
3	d → a c
4	b a → c e
5	e a → b c
6	a → c
7	b c → e
8	e c →b
9	b → e
10	e → b

In addition to the implications, the association rules with different levels of confidence are also generated in the following way. The intents of the concepts are mapped into the closed itemsets. The frequency of their occurrence is the length of the extent.

The concepts which are frequent i.e., those that satisfy the minimum support, are called frequent concepts. For each of the intent of frequent concept *l*, generate all non-empty subsets of *l*. For every non-empty subset *s* of *l*, output the rule "*s*→(*l* - *s*)", if confidence of the rule is greater than the minimum confidence threshold [22]. The rules for the context in Table 1 are given in Table 7.

Table 7: Association rules generated using FCA.

Rule	Confidence
a → c	100%
b → c e	66.67%
b → e	100%
b c → e	100%
b e →c	66.67%
c → a	66.67%
c → b e	66.67%
c e → b	100%
e → b	100%
e → b c	66.67%

4. Experimental Results

In this dataset, objects are journals (with impact factor > 3.0) and the attributes include the main topics (disciplines covered) in those journals. The formal context of the DBLP dataset is shown in the following Table 8. The journals are listed in Table 9. Table 10shows the disciplines covered in the journals. Table 11 lists the formal concepts generated for the DBLP dataset using "Object Intersections" algorithm.

Table 8: DBLP formal context.

	a	b	c	d	e	f	g	h	i	j
1		X	X							
2	X							X		
3	X			X						
4		X	X							
5							X			

6	X	X	X					X		
7	X	X		X	X		X			
8		X	X							X
9			X							
10	X	X		X						
11	X									
12		X	X							
13	X									
14			X			X	X			
15	X				X					
16	X									
17	X	X	X	X				X	X	
18	X	X	X		X		X		X	

Table 9: List of journals.

No.	Journal
1	Nucleic Acids Research
2	IEEE Transactions on Pattern analysis and Machine Intelligence
3	International Journal of Computer Vision
4	Computer Applications in the Biosciences
5	Journal of Selected Areas in Communications
6	Transactions on Medical Imaging
7	Transactions on Information Theory
8	BMC Bioinformatics
9	Transactions on Neural Networks

10	Journal of Chemical Information and Computer Sciences
11	Transactions on Fuzzy Systems
12	Journal of Computational Chemistry
13	Transactions on Graphics
14	Transactions on Mobile Computing
15	Transactions on Image Processing
16	Pattern Recognition
17	Automatica
18	Information Sciences

Table 10: List of disciplines.

Attribute	Description
a	Artificial Intelligence and Image Processing
b	Computation Theory and Mathematics
c	Computer Software
d	Information Systems and Library and Information Studies
e	Data Format
f	Distributed Computing
g	Communication Technologies
h	Computer Architecture
i	Data and Information Processing
j	Software Testing and Verification and Validation

Table 11: Formal concepts generated from DBLP dataset.

No.	Formal Concepts
C1	(φ, a b c d e f g h i j)
C2	(1 4 6 8 12 17 18, b c)
C3	(2 6 17 18, a h)

C4	(3 7 10 17, a d)
C5	(2 3 6 7 10 11 13 15 16 17 18, a)
C6	(5 7 14 18, g)
C7	(6 17 18, a b c h)
C8	(7, a b d e g)
C9	(1 4 6 7 8 10 12 17 18, b)
C10	(6 7 10 17 18, a b)
C11	(8, b c j)
C12	(1 4 6 8 9 12 14 17 18, c)
C13	(7 10 17, a b d)
C14	(14, c f g)
C15	(7 15 18, a e)
C16	(17, a b c d h i)
C17	(18, a b c e g h)
C18	(7 18, a b e g)
C19	(14 18, c g)
C20	(1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18, φ)

Consider the concept C3 i.e., (2 6 17 18, a h). The intents a, h i.e., the disciplines “Artificial Intelligence and Image Processing”, “Computer Architecture” are covered in the journals 2, 6, 19, 20 (i.e., extent of the concept). In each concept, the intent represents the maximum possible set of disciplines that are included in all the journals of the extent. Concept lattice for the formal context in Table 8 is shown in Figure 2. It is used for the visualization of the concept hierarchies, which shows the generalization and specialization relationships among concepts. Implications are generated from the formal concepts using the “Find Implications” algorithm. The implications generated for the DBLP dataset are listed in Table 12. The implication a g → b e denotes that the journal with disciplines a (Artificial Intelligence and Image Processing) and g (Communication Technologies) also include the disciplines b

(Computation Theory and Mathematics) and e (Data Format). Thus, by generating the implications, the dependencies among the occurrence of different disciplines in various journals can be analyzed. The number of implications generated for the datasets DBLP, Tuberculosis and Zoo are compared. Tuberculosis dataset is a part of consumer health care informatics project of Medical Research Council of South Africa [5]. This dataset considers 21 patients as objects and 13 symptoms of TB as attributes. The zoo dataset is taken from UCI repository of Machine learning. This dataset considers different animals as objects.

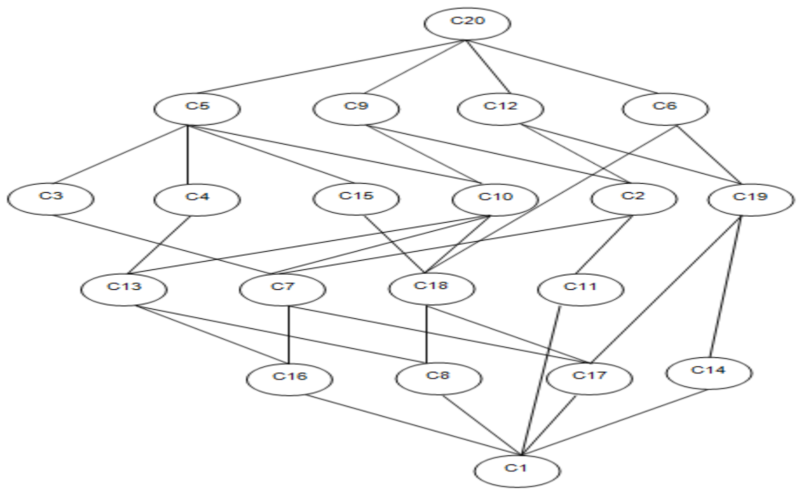


Figure 2: Concept Lattice of DBLP dataset.

Table 12: Implications for DBLP dataset.

No.	Implications
1	a b c → h
2	a b h → c
3	a c → b h
4	b h → a c
5	c h →a b
6	j →b c
7	d → a
8	e →a
9	h → a
10	a g → b e

11	b g → a e
12	e b → a g
13	e g → a b
14	e c → a b g h
15	e h → a b c g
16	g c → a b e h
17	g h → a b c e
18	d c → a b h i
19	d h → a b c i
20	i → a b c d h
21	e d → a b g
22	g d → a b e
23	f → c g

The properties like feathers, predator, aquatic, domestic etc. are considered as attributes. The details of datasets and the number of implications generated are listed in the Table 13. Figure 3 shows the graph for number of rules generated for three datasets at different confidence values.

Table 13: Dataset details.

No.	Dataset	Objects	Attributes	No. of Implications
1	Zoo	101	23	132
2	Tuberculosis	21	13	28
3	DBLP	18	10	23

The time complexity of an algorithm depends on factors such as the size of the input, and the type of operations performed. The time complexity of FCA technique is the overall time complexity of all algorithms used to generate association rules. Formal concepts are generated by using Object Intersection algorithm and then Concepts Cover algorithm is used to build concept lattice

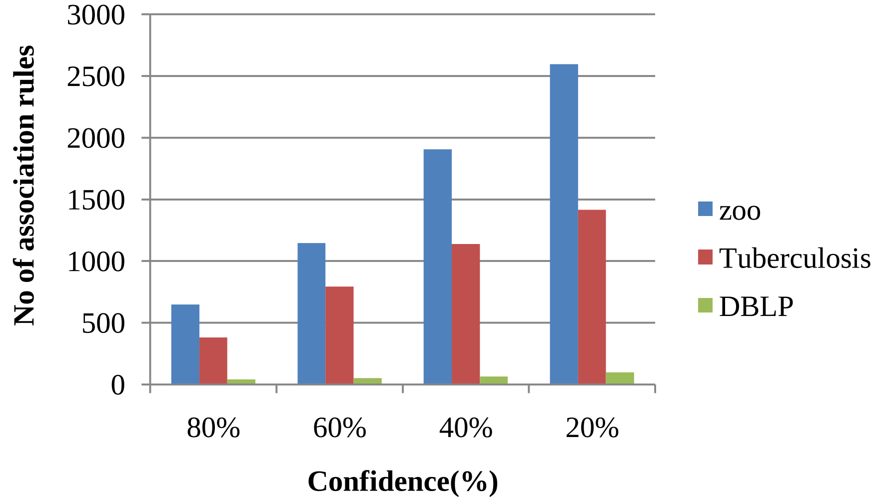


Figure 3: Graph showing number of association rules at different confidence values.

From the generated formal concepts, association rules are generated by using Find Implications algorithm. So, the overall time complexity is the sum of time complexities of Object Intersections algorithm, Concepts Cover algorithm and Find Implications algorithm. The time complexities of algorithms are listed in Table 14.

Table 14: Time complexities of algorithms.

Algorithm	**Time complexity**
Object Intersections Algorithm	O(|G| |C| |M|)
Concepts Cover Algorithm	O(|C| |M| (|G| + |M|))
Find Implications Algorithm	O(|C| k^2 |M| q)

Conclusions

This paper is aimed at providing an understanding on generating association rules using FCA. Concepts are generated using Object Intersections algorithm, ConceptsCover algorithm is used for generating the concept lattice and attribute implications are produced using Find Implications algorithm. Experiments are performed on various datasets obtained from the UCI repository.

Future work will focus on the quality of the generated association rules. Since the time complexity of FCA is more, future work can concentrate on the parallelization of FCA algorithms and new programming environment such as Hadoop.

References

1. Agrawal R., Imielinski T., Swami A. Mining Association Rules between Sets of Items in Large Databases. Proceeding(s) of the ACM SIGMOD International Conference on Management of Data (SIGMOD93), ACMPress.1993: 207-216.
2. Agrawal R., Mannila H., Srikant R., Toivonen H.,Verkamo A. Fast Discovery of Association Rules.Advances in Knowledge Discovery and Data Mining. 1996: 307-328.
3. Agrawal R., Srikant R. Fast Algorithms for Mining Association Rules. Proceedings of the 20th International Conference on Very Large Databases. 1994: 487-499.
4. Alwahaishi S., Martinovic J., Snasel V., Kudelka M. Analysis of the DBLP Publication Classification using Concept Lattices.Digital Enterprise and Information Systems, Springer Berlin Heidelberg. 2011; 132-139.
5. Kumar, C. A. Fuzzy clustering-based formal concept analysis for association rules mining. Applied artificial intelligence. 2012; *26*(3): 274-301.
6. Aswani Kumar Ch., Srinivas S. Mining Associations in Health Care Data using Formal Concept Analysis and Singular Value Decomposition. Journal of Biological Systems, vol. **18**. 2010; 787-807.
7. Carpineto C., Romano G. Concept Data Analysis: Theory and Applications. John Wiley & Sons. 2004.
8. Ceglar A., Roddick J. F. Association mining. ACMComputing Surveys (CSUR). 2002; 38(2): 5.
9. Ch. Aswani Kumar, M. Radvansky, J. Annapurna. Analysis of Vector Space Model, Latent Semantic Indexing and Formal Concept Analysis for Information Retrieval. Cybernetics and Information Technologies.2012; 12(1):34-48.
10. Ch. Aswani Kumar. Knowledge Discovery in Data Using Formal Concept Analysis and Random Projections. International Journal of Applied Mathematics and Computer Science. 2011; 21(4): 745-756.
11. Davey B.A., Priestley M.A. Introduction to Lattices and Order. Cambridge University Press. 2002.
12. Deogun J.S, Raghavan V.V., Sever H. Association Mining and Formal Concept Analysis. In Proceedings of Sixth International Workshop on Rough Sets, Data Mining and Granular Computing. 1998; 335-338.
13. Du Y., Li H.M. Strategy for Mining Association Rules for Web Pages Based on Formal Concept Analysis. Applied Soft Computing.2010; 10(3): 772-783.

14. Jaoua A., Elloumi S. Galois Connection Formal Concept and Galois Lattice in Real Binary Relation. Journal of Systems and Software. 2002; 60(2): 149–163.

15. Kotsiantis S., Kanellopoulos D. Association Rules Mining: A Recent Overview. GESTS International Transactions on Computer Science and Engineering. 2006; 31(2): 71-82.

16. Maddouri M. A. Formal Concept Analysis Approach to Discover Association Rules from Data. In: Proceedings of International Conference on Concept lattices and their applications, 2005, 10-21.

17. Poelmans J., Ignatov D. I, kuznetsov S. O. Formal concept analysis in knowledge processing: A survey on applications. Expert systems with applications. 2013; 40(16): 6538-6560.

18. Priss U. Formal Concept Analysis in Information Science. Annual Review of Information Science and Technology (ARIST). 2007; 40(1): 521-543.

19. Stumme G. Formal Concept Analysis. Handbook on Ontologies. Springer Berlin Heidelberg. 2009; 177-199.

20. Stumme, G. Efficient Data Mining Based on Formal Concept Analysis.International Conference on Database and Expert Systems Applications. 2002; 534-546.

21. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L. Intelligent Structuring and Reducing of Association Rules with Formal Concept Analysis. In Annual Conference on Artificial Intelligence. Springer Berlin Heidelberg. 2001; 335-350.

22. Toui A.G., Thabet A., Sassi M. Efficient Reduction of Number of Association Rules Using Fuzzy Clustering on the Data. Proceedings of 2nd International Computer Services Inc., Part-II, Springer Berlin Heidelberg.2011; 191-199.

23. Valtchev P.,Missaoul R.,Godin R. Formal Concept Analysis for Knowledge Discovery and Data Mining: The New Challenges.In International Conference on Formal Concept Analysis. 2004; 352-371.

24. Wille R. Restructuring lattice theory: An approach based on hierarchies of concepts. In International Conference on Formal Concept Analysis. 2009; 314-339.

25. Ch. Aswani Kumar. Mining association rules using non-negative matrix factorization and formal concept analysis. Computer Networks and Intelligent Computing. Springer Berlin Heidelberg. 2011; 31-39.

26. Kumar C. A., and S. Srinivas. Concept lattice reduction using fuzzy K-means clustering. Expert systems with applications. 2010; 37(3): 2696-2704.

27. Kumar C. A., M. S. Ishwarya, and Chu Kiong Loo. Formal concept analysis approach to cognitive functionalities of bidirectional associative memory. Biologically Inspired Cognitive Architectures. 2015; 12: 20-33.

28. Shivhare R., Cherukuri A. K. Three-way conceptual approach for cognitive memory functionalities. International Journal of Machine Learning and Cybernetics. 2016; 1-14.

29. Ch A. K, Dias S. M., Vieira N. J. Knowledge reduction in formal contexts using non-negative matrix factorization.Mathematics and Computers in Simulation. 2015; 109: 46-63.

30. Singh P. K., Kumar, C. A. Bipolar fuzzy graph representation of concept lattice. Information Sciences. 2014; *288*: 437-448.

31. Singh P. K., Cherukuri A. K., Li J. Concepts reduction in formal concept analysis with fuzzy setting using Shannon entropy. International Journal of Machine Learning and Cybernetics. 2015; 1-11.

32. Li J., Mei C., Lv Y. Knowledge reduction in real decision formal contexts. Information Sciences. 2012; *189*: 191-207.

33. Li J., Mei C., Lv Y. A heuristic knowledge-reduction method for decision formal contexts. Computers & Mathematics with Applications. 2011; 61(4): 1096-1106.

34. Li J., Mei C., Xu W., Qian, Y. Concept learning via granular computing: a cognitive viewpoint. Information Sciences. 2015; 298: 447-467.

35. Li J., Mei C., Lv, Y. Knowledge reduction in formal decision contexts based on an order-preserving mapping. International Journal of General Systems. 2012; 41(2): 143-161.