Armenian Journal of Mathematics
Volume 8, Number 2, 2016, 86–95

Characterizing trees in property-oriented concept lattices

H. Mao

Abstract. Property-oriented concept lattices are systems of conceptual clusters called property-oriented concepts, which are partially ordered by the subconcept/superconcept relationships. Property-oriented concept lattices are basic structures used in formal concept analysis. In general, a property-oriented concept lattice may contain overlapping clusters and is not to be a tree construction. Additionally, tree-like classification schemes are appealing and are produced by several clustering methods. In this paper, we present necessary and sufficient conditions on input data for the output property-oriented concept lattice to form a tree after one removes its greatest element. After applying to input data for which the associated property-oriented concept lattice is a tree, we present an algorithm for computing property-oriented concept lattices.

Key Words: Property-Oriented Concept Lattice, Tree, Property-Oriented Concept
Mathematics Subject Classification 2000: 68R10, 05C05, 68P05

Introduction

As pointed in [3, 4, 5], huge data sets and various data types lead to new problems and require the development of new types of techniques for modern intelligent data analysis. An important objective of intelligent data analysis is to reveal and indicate diverse non-trivial features or views of a large amount of data. Many techniques in data mining and other fields have been proposed. Each technique focuses on one particular view of the data and discovers a specific domain of knowledge embedded in data.

According to the authors in [1], generating collections of clusters from data is a challenging part of knowledge discovery. Among many methods for performing this task, formal concept analysis (FCA) is becoming increasingly

popular. The main aim of FCA is to extract interesting clusters from tabular data along with a partial order of these clusters. FCA yields diagrams of hierarchically ordered clusters which are lattices and termed as concept lattices.

Lattice theory provides a vocabulary for hierarchical structures and is applied to construct concept lattices (see [3, 4, 5, 6]). There are several types of concept lattices which are derived from a same formal context and reflect various features of the data [3, 6]. Though the results on formal concept lattices introduced by R.Wille in 1982 [11] are much more used and noticed by researchers than that of other concept lattices, the other clustering classification techniques such as property-oriented concept lattice first introduced by Gediga and Düntsch [6] are also important and crucial in the theoretical development FCA.

S. Radelezcki proposed a problem [9] about tree-order for formal concept lattices. It is one of the problems to seek out possible relationships between FCA and the other methods of clustering and classification. Needless to say, this goal requires a long-term effort. In this respect, the author [9] highlights certain important implications of tree construction in FCA. Along this direction, we consider a particular problem in this paper. We present conditions for input data which are necessary and sufficient for the output property-oriented concept lattice to form a tree after removing its greatest element. Similar consequences have already appeared in [1] for characterizing formal concept lattices, but the aforementioned goal has not yet been achieved. We hope our results here help partially to achieve the goal.

It is more important as [1] shows to think about the purposes of clustering, about the types of clusters we wish to construct. We must seek sufficiently rich class of structures for the research on FCA. Hence, looking for algorithms and properties for property-oriented concept lattices is also an important duty for the study of FCA.

We see as [1, 4, 5] that characterizations of trees in formal concept lattices are good to some algorithms in formal concept lattices. We find from [3, 6, 10] that property-oriented concept lattices are equally essential in FCA as formal concept lattices. The notion of property-oriented concept has first been proposed in [6], but without exploring its category-theoretical content. As [3, 6, 10], researches have not searched out an algorithm to construct the property-oriented concept lattices. Hence, our another duty is to find an algorithm to search out all the property-oriented concepts and its property-oriented concept lattice after a formal context generates a tree.

Since there are the intimate relationships between formal concept lattices and property-oriented concept lattices (see [3, 6]), some discussions in this paper are seemly similar to those in [1] for formal concept lattices. In spite of this, we may be assured as [3, 6] that the independent conceptual structures of property-oriented concepts are different from that of formal concepts for

a same formal context. Hence, all the ideas and algorithms in this paper are independent and own their values to exist in FCA.

1 Preliminaries

A very preliminary report on some of the results and notions in this paper has appeared as [3, 4, 5, 6]. In this section, we summarize basic notions of FCA. For detailed information on FCA, we refer to [3, 4, 5, 6]. More details for lattice theory, please see [4, 5, 7].

1.1 Formal context

This subsection summarizes basic notions of formal contexts.

An object-attribute data table describing which objects have what attributes can be identified with a triplet (U, V, R) where U is a non-empty set (of *objects*), V is a non-empty set (of *attributes*), and $R \subseteq U \times V$ is an (*object-attribute*) relation. (U, V, R) is called a *formal context* (simply, *context*). For $x \in U$ and $y \in V$, if $(x, y) \in R$, written as xRy, we say that x has the attribute y, or the attribute y is possessed by an object x.

A context can be easily represented by a *cross table*, i.e., by a rectangular table the rows of which are headed by the object names and the columns headed by the attribute names. A cross in row g and column m means that the object g has the attribute m.

Based on the binary relation R, we can associate a set of attributes with an object. An object $x \in U$ has the set of attributes, $xR = \{y \in V \mid xRy\} \subseteq V$. The set of attributes xR can be viewed as a description of the object x. In other words, object x is described or characterized by the set of attributes xR. Similarly, an attribute y is possessed by the set of objects, $Ry = \{x \in U \mid xRy\} \subseteq U$.

1.2 Property-oriented concept lattice

This subsection presents some notations and properties for property-oriented concept lattices from [3].

For a set of objects $A \subseteq U$ and a set of attributes $B \subseteq V$, we can define a pair data operators, $\Box : 2^U \to 2^V$ and $\Box : 2^V \to 2^U$ as follows: $A^\Box = \{y \in V \mid Ry \subseteq A\}$, $\quad B^\Box = \{x \in U \mid xR \subseteq B\}$.

The same symbol is again used for both operators.

We can define a pair data operators, $\diamond : 2^U \to 2^V$ and $\diamond : 2^V \to 2^U$ as follows:

$A^\diamond = \{y \in V \mid Ry \cap A \neq \emptyset\} = \bigcup\limits_{x \in A} xR$, $\quad B^\diamond = \{x \in Y \mid xR \cap B \neq \emptyset\} = \bigcup\limits_{y \in B} Ry$.

Again, the same symbol is used for both operators.

A pair $(A, B), A \subseteq U, B \subseteq V$, is called a *property-oriented formal concept* if $A = B^{\Box}$ and $B = A^{\diamond}$. The set of objects A is referred to as the *extension* of the concept (A, B), and the set of attributes B is referred to as the *intension*. If an attribute is possessed by an object in A, then the attribute must be in B. Moreover, only attributes in B are possessed by objects in A.

For two property-oriented formal concepts (A_1, B_1) and (A_2, B_2), we say that (A_1, B_1) is a *sub-concept* of (A_2, B_2), and (A_2, B_2) is *super-concept* of (A_1, B_1), in notation, $(A_1, B_1) \leq (A_2, B_2)$, if and only if $A_1 \subseteq A_2$, or equivalently, if and only if $B_1 \subseteq B_2$.

The family of all property-oriented formal concepts $\mathfrak{P}(U, V, R)$ is a complete lattice, termed as *property-oriented formal concept lattice* (see [12]). The meet $\wedge$ and the join $\vee$ of the property-oriented formal concept lattice are defined by

$(A_1, B_1) \wedge (A_2, B_2) = ((A_1 \cap A_2), (B_1 \cap B_2)^{\Box\diamond})$,

$(A_1, B_1) \vee (A_2, B_2) = ((A_1 \cup A_2)^{\diamond\Box}, (B_1 \cup B_2))$.

For simplicity, a property-oriented formal concept lattice and a property-oriented formal concept in this paper is called a property-oriented concept lattice and a property-oriented concept respectively.

Consider the operator $^{\diamond\Box} : 2^U \to 2^U$, which is a closure operator on 2^U [1, 3]. The properties of this operator are: for $A, A_1, A_2 \subseteq U$,

(p1) $U^{\diamond\Box} = U$,

(p2) $A \subseteq A^{\diamond\Box}$,

(p3) $A^{\diamond\Box} = A^{\diamond\Box\diamond\Box}$,

(p4) $A_1 \subseteq A_2 \Rightarrow A_1^{\diamond\Box} \subseteq A_2^{\diamond\Box}$.

Let $(U,^{\diamond\Box}) = \{A^{\diamond\Box} \mid A \subseteq U\}$. Then $(U,^{\diamond\Box})$ is a closure system with the following properties:

(pi) $\emptyset \in (U,^{\diamond\Box})$,

(pii) for a non-empty set $A \in (U,^{\diamond\Box})$, we have: $(A, A^{\diamond}) = \bigvee\{(\{x\}^{\diamond\Box}, \{x\}^{\diamond}) \mid x \in A\}$.

The property (pii) shows that the pair $(\{x\}^{\diamond\Box}, \{x\}^{\diamond})$ is a property-oriented concept to generate other property-oriented formal concepts in the lattice.

Since property-oriented concept lattices are complete lattices, we find that each property-oriented concept lattice $\mathfrak{P}(U, V, R)$ has both the greatest (U, V) and the least $(\emptyset, \emptyset)$.

2 Trees in property-oriented concept lattices

This section interests in property-oriented concept lattices corresponding to trees. Trees are usually defined as undirected graphs that are acyclic and connected (cf. [2, 8]). Since we are going to identify trees in particular ordered sets, we deal with trees as follows. A finite partially ordered set $(P, \preceq)$ will be called a *tree* if for each $a, b \in P$:

(Ti) there is an infimum of a and b in $(P, \preceq)$, and

(Tii) there is a supremum of a and b in $(P, \preceq)$ if and only if a and b are comparable (i.e., if and only if $a \preceq b$ or $b \preceq a$).

For easily comparing the relationships between our results and the already existed apprehensions in formal concept lattices, the definition of tree here is similar to [1] and [2]. Hence, we can state that the definition of tree here is much more direct and natural.

Following the above, we can express that the whole property-oriented concept lattice is a tree if and only if it is linearly ordered, which is not worthwhile observation because linear trees are a degenerate form of trees and therefore not interesting. Now, we focus our attention on trees which are important for property-oriented concept lattices. In what follows, we explore the conditions to make $\mathfrak{P}^{\curlywedge}(U, V, R)$ a tree where $\mathfrak{P}^{\curlywedge}(U, V, R)$ denotes $\mathfrak{P}(U, V, R) - \{(U, V)\}$.

2.1 Formal contexts generating trees

It is easily check the following statements:

(3.1.1) $\mathfrak{P}^{\curlywedge}(U, V, R)$ has the same partial order as $\mathfrak{P}(U, V, R)$.

(3.1.2) A tree structure for $\mathfrak{P}^{\curlywedge}(U, V, R)$ is the simplest among all the structures which $\mathfrak{P}^{\curlywedge}(U, V, R)$ can possibly possess.

(3.1.3) In a tree structure, we can not find redundant concepts. Hence, if we search out a tree structure of $\mathfrak{P}^{\curlywedge}(U, V, R)$, then we find all the concepts with no redundant elements so as to save time and spaces during our searching process.

The following assertion characterizes $\mathfrak{P}^{\curlywedge}(U, V, R)$ to be a tree in terms of intensions of property-oriented concepts.

Theorem 1 *labelth.1 Let (U, V, R) be a formal context. Then $\mathfrak{P}^{\curlywedge}(U, V, R)$ is a tree if and only if, for any concepts $(A, B), (C, D) \in \mathfrak{P}(U, V, R)$, at least one of the following is true:*

(i) $B \subseteq D$ or $D \subseteq B$,

(ii) $B \cup D = V$.

Proof. Let $\mathfrak{P}^{\curlywedge}(U, V, R)$ be a tree.

Suppose that (ii) is not satisfied. Then, by the definition of tree, it is easily seen that (i) must be satisfied.

Suppose that (i) is not satisfied for some $(A,B),(C,D) \in \mathfrak{P}(U,V,R)$. By the definition of $\leq$ in Section 1.2, we have $(A,B) \not\leq (C,D)$ and $(C,D) \not\leq (A,B)$. Then both (A,B) and (C,D) belong to $\mathfrak{P}^{\curlywedge}(U,V,R)$. This implies that $(A,B) \vee (C,D)$ is in $\mathfrak{P}(U,V,R) = \mathfrak{P}^{\curlywedge}(U,V,R) \cup \{(U,V)\}$. Combining with Section 1.2, we obtain that $\mathfrak{P}(U,V,R)$ is a complete lattice with (U,V) as the greatest element. Therefore, we receive $(A,B) \vee (C,D) = ((A \cup C)^{\diamond\square}, (B \cup D)) = (U,V)$. So, $B \cup D = V$ holds. Thus, the item (ii) is satisfied.

Conversely, if any two elements in $\mathfrak{P}(U,V,R)$ are comparable, that is, any two elements satisfy (i), then we obtain that $\mathfrak{P}(U,V,R)$ is a chain. Thus, the needed result is accepted.

Let $(A,B),(C,D) \in \mathfrak{P}^{\curlywedge}(U,V,R)$ such that (A,B) and (C,D) are incomparable. Such (A,B) and (C,D) cannot satisfy (i), i.e., we have $B \cup D = V$. That is to say, (A,B) and (C,D) satisfy the item (ii). Using the definition of $\mathfrak{P}(U,V,R)$, we obtain that $(A,B) \vee (C,D)$ is the greatest element in $\mathfrak{P}(U,V,R)$. Thus, (A,B) and (C,D) do not have a supremum in $\mathfrak{P}^{\curlywedge}(U,V,R)$ because $\mathfrak{P}^{\curlywedge}(U,V,R)$ is $\mathfrak{P}(U,V,R) - \{(U,V)\}$. Therefore, we find that $\mathfrak{P}^{\curlywedge}(U,V,R)$ is a tree.$\square$

Can we find $\mathfrak{P}^{\curlywedge}(U,V,R)$ to be a tree from (U,V,R) before computing the set of all concepts? We will give a positive answer. For convenient, we need the following notion.

Definition 1 *Let (U,V,R) be a formal context. We say that (U,V,R) generates a tree if $\mathfrak{P}^{\curlywedge}(U,V,R)$ is a tree.*

Theorem 2 *Let (U,V,R) be a formal context. Then (U,V,R) generates a tree if and only if, for any attributes $y_1, y_2 \in U$, at least one of the following conditions is true:*

(i) $\{y_1\}^{\diamond} \subseteq \{y_2\}^{\diamond}$,

(ii) $\{y_2\}^{\diamond} \subseteq \{y_1\}^{\diamond}$,

(iii) $\{y_1\}^{\diamond} \cup \{y_2\}^{\diamond} = V$.

Proof. Suppose that (U,V,R) generates a tree, i.e. $\mathfrak{P}^{\curlywedge}(U,V,R)$ is a tree. Each pair of the form $(\{x\}^{\diamond\square}, \{x\}^{\diamond})$ is a property-oriented concept in the $\mathfrak{P}(U,V,R)$ by Subsection 1.2. We find by virtue of Theorem 1 that the items (i)-(iii) are special cases in Theorem 1. Therefore, the items (i)-(iii) are accepted in light of Theorem 1.

Conversely, suppose that (U,V,R) does not generate a tree. Thus, there are incomparable concepts $(A,B),(C,D) \in \mathfrak{P}(U,V,R)$ such that $(A,B) \vee (C,D)$ in $\mathfrak{P}(U,V,R)$ does not equal to (U,V). This implies that $B \nsubseteq D, D \nsubseteq B, B \cup D \subset V$(i.e. $B \cup D \subsetneq V$), and $((A \cup C)^{\diamond\square}, B \cup D) \in \mathfrak{P}(U,V,R)$.

Hence, we obtain $A \nsubseteq C$ and $C \nsubseteq A$. However, $A \nsubseteq C$ means the existence of $y_1 \in A \setminus C$, and meanwhile, $C \nsubseteq A$ means the existence of $y_2 \in C \setminus A$. Combining (p4) and $y_1 \in A$, we receive $\{y_1\}^{\diamond\Box} \subseteq A^{\diamond\Box}$. In addition, $(A^{\diamond\Box}, B) \in \mathfrak{P}(U, V, R)$ shows $A^{\diamond\Box} = A$. Furthermore, the concept $(\{y_1\}^{\diamond\Box}, \{y_1\}^{\diamond})$ satisfies $(\{y_1\}^{\diamond\Box}, \{y_1\}^{\diamond}) \leq (A, B)$. Therefore, we attain $\{y_1\}^{\diamond} \subseteq B$. Similarly, we obtain $\{y_2\}^{\diamond} \subseteq D$. In fact, we produce $\{y_1\}^{\diamond} \cup \{y_2\}^{\diamond} \subseteq B \cup D \subset V$. So $\{y_1\}^{\diamond} \cup \{y_2\}^{\diamond} \neq V$ holds.

Next we prove $\{y_1\}^{\diamond} \nsubseteq \{y_2\}^{\diamond}$ and $\{y_2\}^{\diamond} \nsubseteq \{y_1\}^{\diamond}$.

$y_1 \in A \setminus C$ asks $y_1 R \nsubseteq D$. Otherwise, it follows $y_1 \in D^{\Box} = C$, a contradiction to $y_1 \in A \setminus C$. On the other hand, $y_2 \in C \setminus A \subseteq C = D^{\Box}$ makes $y_2 R \subseteq D$. Combining $\{y_1\}^{\diamond} = y_1 R$ and $\{y_2\}^{\diamond} = y_2 R$ with the above two hands, we demonstrate $\{y_1\}^{\diamond} \nsubseteq \{y_2\}^{\diamond}$.

Analogously, we can demonstrate $\{y_2\}^{\diamond} \nsubseteq \{y_1\}^{\diamond}$.

Summing up, we may be assured that if (U, V, R) does not generate a tree, then there are $y_1, y_2 \in U$ such that none of (i)-(iii) is satisfied.□

2.2 Algorithms for trees in property-oriented concept lattices

Trees in concept lattices, as they were introduced in Section 1, Subsection 2.1 and [1, 4, 5, 7], can be computed by algorithms. Currently, there have been proposed several algorithms for computing formal concepts (see [4, 5]), though seldom are for computing property-oriented concepts according to my knowledge. In spite of this, all these algorithms can be referred to compute any of concept lattices for FCA in view of [7]. R.Bělohlávek et al indicate [1] that for some algorithms, the searching structure should be efficient because the tests of presence of a concept between the found concepts influences the overall efficiency of the procedure. The searching structure is usually implemented as searching a tree. In fact, some of the algorithms for FCA get simplified in the case of contexts generating tree constructions (see [1, Algorithm 1]). This section is to present an algorithm to compute all the property-oriented concepts for a formal context generating a tree.

The sketch of the algorithm for computing $\mathfrak{P}^{\curlywedge}(U, V, R)$ is shown in the following if (U, V, R) generates a tree.

Let $U = \{y_1, y_2, \ldots, y_t\}$. After computing $\{y_j\}^{\diamond}\,(j = 1, 2, \ldots, t)$, we set

$$\mathcal{B} = \left(\left\{ \{y_j\}^{\diamond} \middle| \, j = 1, 2, \ldots, t \right\}, \subseteq \right).$$

Evidently, $\mathcal{B}$ is a poset.

We assert that $(\mathcal{B}, \subseteq)$ is isomorphic to the poset $\mathfrak{P}^{\curlywedge}(U, V, R) \setminus \{(\emptyset, \emptyset)\}$, and $\mathfrak{P}^{\curlywedge}(U, V, R) \setminus \{(\emptyset, \emptyset)\}$ is uniquely determined by $(\mathcal{B}, \subseteq)$. The reasons are as follows.

(α) By the definition of property-oriented concept, we only need to consider the property of the intension D of a concept (C, D) if we detect the property of (C, D).

(β) We find that for a non-empty set $A \subseteq U, (A, A^{\diamond}) \in \mathfrak{P}(U, V, R)$ if and only if

$$(A, A^{\diamond}) = \bigvee\{(\{x\}^{\diamond\Box}, \{x\}^{\diamond}) \mid x \in A\} = ((\bigcup_{x\in A}\{x\}^{\diamond\Box})^{\diamond\Box}, \bigcup_{x\in A}\{x\}^{\diamond}),$$

equivalently to say, if and only if $A^{\diamond} = \bigcup\limits_{x\in A}\{x\}^{\diamond}$, and $A = (\bigcup\limits_{x\in A}\{x\}^{\diamond\Box})^{\diamond\Box}$.

(γ) According to Theorem 2 and the tree structure of $\mathfrak{P}^{\curlywedge}(U, V, R)$, we obtain $\{y_i\}^{\diamond} \subseteq \{y_j\}^{\diamond}$, or $\{y_j\}^{\diamond} \subseteq \{y_i\}^{\diamond}$, or $\{y_i\}^{\diamond} \cup \{y_j\}^{\diamond} = V$, for any $y_i, y_j \in U$.

Assume $(A, B) \in \mathfrak{P}^{\curlywedge}(U, V, R) \setminus \{(\emptyset, \emptyset)\}$ and $B \notin \mathcal{B}$. Then by the construction of property-oriented concept lattice $\mathfrak{P}(U, V, R)$, we attain $A^{\diamond} = B = \bigvee\limits_{a\in A}\{a\}^{\diamond}$. It is no harm to suppose $A = \{y_1, \ldots, y_n\}$. So, $B = \bigcup\limits_{j=1}^{n}\{y_j\}^{\diamond}$ holds. If $p, q \in \{1, \ldots, n\}$ and $\{y_p\}^{\diamond} \nsubseteq \{y_q\}^{\diamond} \nsubseteq \{y_p\}^{\diamond}$, then $B = V$. This is a contrary to $(A, B) \in \mathfrak{P}^{\curlywedge}(U, V, R)$. Therefore, for any $p, q \in \{1, \ldots, n\}$, we may be assured $\{y_p\}^{\diamond} \subseteq \{y_q\}^{\diamond}$ or $\{y_q\}^{\diamond} \subseteq \{y_p\}^{\diamond}$. Considering with [6], we can receive $(\left\{\{y_j\}^{\diamond} \middle| j = 1, \ldots, n\right\}, \subseteq)$ to be a chain (or say, a linearly order). Furthermore, we obtain $y_m \in A$ satisfying $B = \{y_m\}^{\diamond}$.

(δ) Since the reason (γ) hints $B \in \mathcal{B}$, for any $(A, B) \in \mathfrak{P}^{\curlywedge}(U, V, R) \setminus \{(\emptyset, \emptyset)\}$. In addition, $(\{y_j\}^{\diamond\Box}, \{y_j\}^{\diamond}) \in \mathfrak{P}^{\curlywedge}(U, V, R)$ holds in light of (β) where $j = 1, 2, \ldots, t$. Moreover, $\mathcal{B}$ determines $\mathfrak{P}^{\curlywedge}(U, V, R)$ uniquely. Hence, we may easily see that $\mathfrak{P}^{\curlywedge}(U, V, R)$ is isomorphic to $(\mathcal{B}, \subseteq)$.

Consequently, for $\mathfrak{P}^{\curlywedge}(U, V, R)$, we may obtain all of the nodes

$$\{(\{y_j\}^{\diamond\Box}, \{y_j\}^{\diamond}) \mid j = 1, 2, \ldots, t\} \cup \{(\emptyset, \emptyset)\}$$

and its diagram. Furthermore, we produce $\mathfrak{P}(U, V, R)$ because $\mathfrak{P}^{\curlywedge}(U, V, R)$ is produced by removing the greatest element (U, V) from $\mathfrak{P}(U, V, R)$.

Therefore, if (U, V, R) generates a tree, we only focus on the property of $\{y_j\}^{\diamond}$ for every $y_j \in U$ $(j = 1, 2, \ldots, |U|)$ when we compute all the property-oriented concepts for $\mathfrak{P}(U, V, R)$

Actually, $\{y_j\}^{\diamond}$ is easily obtained from the cross table of context (U, V, R). This fact demonstrates that if a formal context generates a tree, then the algorithm for FCA is designed faster and simpler to implement.

3 Conclusions

In this paper, we presented conditions for input data for FCA which are necessary and sufficient for the output property-oriented concept lattice to form a tree after one removes its greatest element. We present an algorithm for FCA different from [1, Algorithm 1]. Since trees are the most common structures which appear in traditional clustering and classification, our future research will focus on establishing connections between FCA and other clustering and classification methods.

Acknowledgment

This research is granted by NSF of China (61572011) and NSF of Hebei Province (A2013201119).

References

[1] R. Bělohlávek, B.D. Baets, J. Outrata, and V. Vychodil, *Characterizing trees in concept lattices*, International J. of Uncertainty, **16**(2008), 1-15.

[2] J.A. Bondy and U.S.R. Murty, *Graph Theory with Applications*. Elsevier Science Publishing Co. Inc., New York, 1976.

[3] Y. Chen and Y. Yao,*A multiview approach for intelligent data analysis based on data operators*, Information Sciences, **178**(2008), 1-20.

[4] B.A. Davey and H.A. Priestley, *Introduction to Lattices and Order*, 2nd. ed. Cambridge University Press, Cambridge, 2003.

[5] B. Ganter and R. Wille,*Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin, 1999.

[6] G. Gediga and I. Düntsch, *Modal-style operators in qualitative data analysis*, Proceedings of the 2002 IEEE International Conference on Data Mining, 2002, pp.155-162.

[7] G. Grätzer, *General Lattice Theory*, 2nd. ed. Birkhäuser-Verlag, Basel, 1998.

[8] D. König, *Theory of finite and infinite graphs*, translated by R.McCoart with commentary by W.T.Tutte, Birkhäuser, Boston, 1990.

[9] S. Radelezcki, *Problem 14, in: Some open problems in formal concept analysis*, Problems Presented at ICFCA 2006 in Dresden. `http://www.upriss.org.uk/fca/problems06.pdf`

[10] X. Wang and W. Zhang, *Relations of attribute reduction between object and property oriented concept lattices*, Knowledge-Based Systems, **21**(2008), 398-403.

[11] R. Wille,*Restructuring lattice theory: an approach based on hierarchies of concepts*, in: Ivan Rival(ed.), Ordered Sets. Dordecht-Boston, Reidel, 1982, 445-470.

[12] Y. Yao,*Concept lattices in rough set theory*, in Proceedings of Annual Meeting of the North American Fuzzy Information Processing Society(NAFIPS'04), 2004, 796-801.

Hua Mao
Department of Mathematics, Hebei University,
Baoding 071002, China.
yushengmao@263.net

Please, cite to this paper as published in
Armen. J. Math., V. **8**, N. 2(2016), pp. 86–95