J. Indones. Math. Soc.
Vol. 23, No. 1 (2017), pp. 13–26.

FORMAL CONCEPT ANALYSIS WITH CONSTRAINTS BY EM OPERATORS

Hua Mao

Department of Mathematics, Hebei University, Baoding 071002, China

yushengmao@263.net

Abstract. Formal concept analysis is a method of exploratory data analysis that aims at the extraction of natural cluster from object-attribute data tables. We present a way to add user's background knowledge to formal concept analysis. The type of background knowledge we deal with relates to relative importance of attributes in the input data.

We introduce EM operators which constrain in attributes of formal concept analysis. The main aim is to make extraction of concepts from the input data more focused by taking into account the background knowledge. Particularly, only concepts which are compatible with the constraint are extracted from data. Therefore, the number of extracted concepts becomes smaller since we leave out non-interesting concepts. We concentrate on foundational aspects such as mathematical feasibility and computational tractability.

Key words and Phrases: Formal concept analysis, EM operator, implication, interesting concept,interesting attribute.

Abstrak. Analisis konsep formal adalah suatu metode analisis data eksplorasi yang bertujuan untuk pengekstraksian klaster alami dari tabel data beratribut objek. Kami menyajikan suatu cara untuk menambahkan latar belakang pengetahuan pengguna untuk analisis konsep formal. Tipe latar belakang pengetahuan yang kami perhatikan adalah tipe yang berkaitan untuk kepentingan relatif dari atribut di data input.

Kami memperkenalkan operator EM yang membatasi atribut analisis konsep formal. Tujuan utama adalah untuk membuat ekstrak konsep dari data input menjadi lebih fokus dengan mengkaitkan ke latar belakang pengetahuan. Secara khusus, hanya konsep yang sesuai dengan batsan yang diekstrak dari data. Sehingga banyaknya konsep yang diekstrak menjadi semakin kecil karena kita mengabaikan konsep yang tidak menarik. Kami memfokuskan pada aspek dasar seperti feasible secara matematika dan dapat dilacak dengan perhitungan.

2000 Mathematics Subject Classification: 68P15, 68P05.
Received: 24-07-2015, revised: 02-02-2017, accepted: 02-02-2017.

Kata kunci: Analisis konsep formal, operator EM, implikasi, konsep yang menarik, atribut yang menarik.

1. INTRODUCTION

Formal concept analysis (FCA) is a method of data analysis and visualization which deals with input data in the form of a table describing objects, their attributes, and their relationship (cf. [10] and [12]). As [8] points, FCA is proved to be useful for knowledge extraction form and visualization of binary data-sets in various application domains such as organization of Web search results into a hierarchical structure of concepts based on common topics, information retrieval, and so on (see [8-10, 12-17]). In addition, the ideas of hierarchy order and logic in FCA are applied to engineer and some other fields (see [11, 18, 19]).

The authors [6] indicate that a distinguishing feature of FCA is an inherent integration of three components: discovery of clusters (so-called formal concepts) in data, discovery of data dependencies (so-called attribute implications) in data, and visualization of formal concepts and attribute implications by a single hierarchical diagram (so-called concept lattice). In the basic setting, people often suppose that no further information is supplied at the input except for the data table. However, it is often the case that there is an additional information available in the form of a constraint (requirement) specified by a user. In such a case, one is not interested in all the outputs but only in those which satisfy the constraint. The other outputs may be left out as non-interesting. This way, the number of outputs is reduced by focusing on the "interesting ones".

We are well known that classification is a way of categorising the data (records) for an attribute. The choice of classification system is critical to information displayed by a map. Classification can be used to enhance the information or to deliberately mislead. Attributes can use different classifications for the same data to change the nature of the display, this can be achieved in some different ways (see [20]).

Therefore, combining the above indications with the ideas in [10, 12, 21, 22], we will believe that roughly speaking, constrain attributes belong to the field of attribute classification. A reader can find examples of using constraints in data mining from all the references such as [1-5, 7, 9, 13, 23-25, 26].

For FCA, comparing the results from no constraints on input data (see [10] and [12]) with ones from constraints by some relations on input data (e.g. [1, 2, 11, 23-25]), we may easily state that for FCA, the case with no constraints on input data can be thought as a special one constrained by some relation such as equivalence relations. Following our comparisons from the results in [1, 2, 24] and [3], we may believe that every equivalence relation is a closure operator.

The discussion in [7] instructs that though some researchers, for example the authors in [3], reveal some properties of FCA with constraints by closure operators, it is still valuable to consider FCA with constraints by other operators. In other

words, to find a simpler operator than a closure operator to constraint on input data is always the pursue of researchers, since the pursued result will lead to a wider application of FCA.

Since we can find "interesting objects" by dual way to constraint attributes, this paper pay attention to explore constraint relations on attributes.

In this paper, we provide an EM operator which is simpler than closure operator. Constrained with EM operator on attributes, we find some properties of S-concepts (that is, "interesting formal concepts"). With the aid of Ganter's NextClosure Algorithm, we discover some relationships between EM operator and "interesting intents". After that, we present two algorithms (including NextEM Algorithm) to search S-intents of a context. Using EM operator and "interesting intents", we describe "interesting" implications between attributes. Our approach is theoretically and computationally tractable and covers several interesting forms of constraints.

The rest of this paper is organized as follows. Section 2 provides preliminaries for FCA. Section 3 presents our approach, theoretical foundations, and the idea of algorithms. Section 4 concludes this paper.

2. Preliminaries

In this section, we summarize basic notions of FCA. For more detail, please, refer to [8, 10, 12]. Additionally, the definition of a closure operator on a set, please, see [3, 10, 12].

In what follows, $A \subseteq B$ means that A is a subset of B; $A \subset B$ means $A \subseteq B$ and $A \neq B$.

Definition 1 ([12, pp.17-20 & 10]) A *formal context* (X, Y, I) consists of two sets X and Y and a relation I between X and Y. The elements of X are called the *objects* and the elements of Y are called the *attributes* of (X, Y, I) . In order to express that an object g is in a relation I with an attribute m, we write gIm. For a set $A \subseteq X$ and a set $B \subseteq Y$, we define $A' := \{m \in Y \mid gIm \text{ for all } g \in A\}$ and $B' := \{g \in X \mid gIm \text{ for all } m \in B\}$.

A *formal concept* of (X, Y, I) is a pair (A, B) with $A \subseteq X, B \subseteq Y, A' = B$ and $B' = A$. We call A the *extent* and B the *intent* of the concept (A, B).

In what follows, (X, Y, I) denotes a formal context, $\mathcal{B}(X, Y, I)$ is the *concept lattice* of (X, Y, I).

Lemma 1 ([12, p.18 &10]) Let (X, Y, I) be a context. If B, B_1, B_2 are sets of attributes, then,

(1) $B_1 \subseteq B_2 \Longrightarrow B_2' \subseteq B_1'$; (2) $B \subseteq B''$; (3) (B', B'') is a concept.

The ideas regarding relative importance of attributes, as described informally before, can be approached in the framework of FCA as follows.

Definition 2 ([3 &12, pp.79-81; 10]) *An attribute implication* over a set Y of attributes is an expression $A \Rightarrow B$, where $A, B \subseteq Y$ are sets of attributes. $A \Rightarrow B$ is *true* in a set $M \subseteq Y$, written as $M \models A \Rightarrow B$, if the following condition is satisfied: if $A \subseteq M$, then $B \subseteq M$.

A set $M \subseteq Y$ is called a *model* of a set T of implications if, for each $A \Rightarrow B \in T, M \models A \Rightarrow B$. Let $Mod(T)$ denote the set of all models of T.

In fact, from [3], we know that a formal concept $(C, D) \in \mathcal{B}(X, Y, I)$ satisfies $A \Rightarrow B$ if and only if $D \models A \Rightarrow B$.

Remark 1 The authors [4-6] provide AD-formulas between attributes respectively. Their motivations provided AD-formulas are similar to that of implications between attributes.

Implications and AD-formulas are introduced and studied between attributes (cf. [3-6, 8, 10, 12]). Belohlavek and Vychodil [4] point out that for $A, B, M \subseteq Y$, we have $M \models A \sqsubseteq B$ if and only if $\overline{M} \models B \Rightarrow A$, where $\overline{M} = Y - M$, $A \sqsubseteq B$ is an AD-formula. Hence, we only need to put our effort on implications between attributes.

3. Constrained Attributes

We are well known from [3] that selecting "interesting" concepts from $\mathcal{B}(X, Y, I)$ needs to be accompanied by a criterion of what is interesting. Such a criterion can be seen as a constraint and depends on particular data and application. Therefore, the constraint should be supplied by a user along with the input data (X, Y, I). One way to specify "interesting concepts" is to focus on concepts whose sets of attributes are "interesting". This seems to be natural because "interesting concepts" are determined by "interesting attributes". In this section, we develop this idea provided that the selected sets of attributes which are taken as "interesting" form a system on Y determined by an EM operator.

3.1. Interesting Formal Concepts.

We start by a definition of EM operator and summarize some interesting sets of attributes using EM operators.

Definition 3 (1) Let Y be a set of attributes and $S : 2^Y \to 2^Y$ be an operator on Y. S is called an *EM operator* if S satisfies the following conditions:
(s1) $A \subseteq S(A)$. (extensive)
(s2) $A \subseteq B$ implies $S(A) \subseteq S(B)$. (monotony)

(2) Let S be an EM operator. A set $B \subseteq Y$ of attributes is called an *S-interesting set of attributes* (shortly, a set of *S-attributes*) if $B = S(B)$. Let
$\mathcal{B}_S(X,Y,I) = \{(A,B) \in 2^X \times 2^Y | A' = B, B' = A, B = S(B)\}$,
and $Int_S(X,Y,I) = \{B \subseteq Y \mid \text{there is } A \subseteq X \text{ such that } (A,B) \in \mathcal{B}_S(X,Y,I)\}$.

Each $(A,B) \in \mathcal{B}_S(X,Y,I)$ is called an *S-interesting concept (S-concept)*; $B \in Int_S(X,Y,I)$ is called an *S-interesting intent (S-intent).*

The following example shows the existence of EM operators.

Example 1 Let $Y = [0,1]$. Define $S : 2^Y \to 2^Y$ as $x \mapsto [0, \sin(\frac{\pi}{2}x)]$, $A \mapsto \bigcup\limits_{x \in A} S(x)$ for any $x \in Y$ and $A \subseteq Y$. It is easily seen that S is an EM operator.

Actually, in Example 1, if $x = \frac{1}{2}$, then $S(x) = [0, \sin(\frac{\pi}{2}\frac{1}{2})]$, $S(S(x)) = \bigcup\limits_{x \in [0, \frac{\sqrt{2}}{2}]} [0, \sin(\frac{\pi}{2}x)] \supset S(x)$. Thus, S is not a closure operator. This fact illustrates that not every EM operator is closure.

Comparing the definitions of a closure operator and an EM operator, we may express that a closure operator is an EM operator.

Thus, we may state that it is valuable to consider the context which its attributes are constrained by EM operators because user's background knowledge need the case to be happened sometimes. Therefore, we may confirm that our discussions in this paper is different from the ones that are constrained attributes by closure operators.

Remark 2 For a given set $B \subseteq Y$ of attributes and an EM operator S, $S(B)$ can be seen as a set of S-attributes containing B. Thus, S is an operator describing which attributes must be added to a set of attributes to make it interesting.

We now focus on the computational aspects of generating all S-concepts. In the sequel, we will show that $\mathcal{B}_S(X,Y,I)$ can be computed with the assistance of Ganter's *NextClosure algorithm* (see [12, pp.66-68]). For this purpose, the first is to combine together two operators: $''$ (operator induced by the Galois connection given by a formal context (X,Y,I)) and S (operator specifying intersting sets of attributes) in the following:

for any $B \subseteq Y$, we define sets B_i ($i \in \mathbb{N}_0$) and $\mathcal{S}(B)$ of attributes as:

$$B_i = \begin{cases} B, & \text{if } i = 0 \\ S(B''_{i-1}), & \text{if } i \geq 1 \end{cases}$$

$$\mathcal{S}(B) = \bigcup_{i=1}^{\infty} B_i. \qquad (*)$$

The second is to show technical insight and crucial properties.

Theorem 1 Let (X,Y,I) be a formal context with $|Y| < \infty$, $S : 2^Y \to 2^Y$ be an EM operator on Y, and $\mathcal{S}$ be defined by $(*)$. Then $\mathcal{S}$ is an EM operator such that for any $B \subseteq Y$, $B = \mathcal{S}(B)$ if and only if $B \in Int_S(X,Y,I)$.

Proof The first is to prove that $\mathcal{S}$ is an EM operator.

In view of (s1) and Lemma 1, we may easily obtain $B_0 = B, B_1 = S(B_0'') \supseteq B_0'' \supseteq B_0$. Repeat this argumentation, we receive $B = B_0 \subseteq B_0'' \subseteq S(B_0'') = B_1 \subseteq B_1'' \subseteq S(B_1'') = B_2 \subseteq \ldots \subseteq B_{i-1} \subseteq B_{i-1}'' \subseteq S(B_{i-1}'') = B_i \subseteq \ldots$ for each $i \in \mathbb{N}_0$.

Thus, considering with $|Y| < \infty$, we may obtain $B \subseteq \mathcal{S}(B)$. Hence, $\mathcal{S}$ is extensive.

Let $A \subseteq B \subseteq Y$. Then, there are $A_0 \subseteq B_0$ and $A_1 = S(A_0'') \subseteq S(B_0'') = B_1$ since Lemma 1 and S satisfies (s2). Repeat this argumentation, in view of the finite of Y, after finite steps, we may obtain $\mathcal{S}(A) \subseteq \mathcal{S}(B)$. That is, $\mathcal{S}$ is monotony.

The second is to prove: $B = \mathcal{S}(B)$ if and only if $B \in Int_S(X, Y, I)$.

We will fulfill by the following ($\Rightarrow$) part and ($\Leftarrow$) part.

($\Rightarrow$) Let $B = \mathcal{S}(B)$.

Then, we obtain $B \subseteq B'' \subseteq S(B'') \subseteq \mathcal{S}(B) = B$. So, it follows $B = B'' = S(B)$. Thereby, in virtue of Lemma 1, Definition 1 and Definition 3, we decide $B \in Int_S(X, Y, I)$.

($\Leftarrow$) Let $B \in Int_S(X, Y, I)$.

Then, we obtain $B = B'' = S(B)$ in light of Definition 1 and Definition 3. This reveals $B_0 = B, B_1 = S(B'') = S(B) = B, B_2 = S(B_1'') = S(B), \ldots, B_i = S(B_{i-1}'') = S(B), \ldots$. Finally, by the finite of Y, it follows $\mathcal{S}(B) = B = S(B) = B''$.

Indeed, combining Theorem 1 with the famous *NextClosure operator* (see [12, p.66]), we provide the following viewpoints.

(1) We can use NextClosure algorithm to compute all the intents of (X, Y, I), that is, $Int(X, Y, I) = \{B = B^+ \mid B \subseteq Y\}$ where B^+ denotes the lectically smallest fixed point of closure operator $''$ which is a successor of B.

(2) Considering (1) with Theorem 1, we believe:

$B = \mathcal{S}(B)$ if and only if $B = B^+ = S(B^+)$.

Thereby, we may attain $Int(X, Y, I)$.

(3) No matter to suppose $Int(X, Y, I) = \{B_1^{\#}, B_2^{\#}, \ldots, B_n^{\#}\}$. Actually, for a given EM operator S, $S(B_j^{\#})$ is well defined ($j = 1, \ldots, n$). Hence, we may easily determine the true or false of $B_j^{\#} = S(B_j^{\#})$.

If it is true, then $Int_S(X, Y, I) = Int_S(X, Y, I) \cup \{B_j^{\#}\}$.

Otherwise, if $j < n$, consider $j + 1$ and repeat the above examination. After n steps, we obtain $Int_S(X, Y, I)$.

Let $Y = \{1, 2, \ldots, n\}$ with $n < \infty$ and $g : 2^Y \to 2^Y$ be an EM operator. We define an *lectically smaller* relation $<$ and the relation $<_i$ as [12, p.66]. Define $A \oplus i = g(((A \cap \{1, 2, \ldots, i-1\}) \cup \{i\})'')$ $(i \in Y)$ for any $A \subseteq Y$. We may attain that the lectically smallest g-interesting intent is $g(\emptyset'')$. Sometimes, we can suppose $\emptyset$ as $g(\emptyset'')$ since this will not infect any discussion for searching g-interesting intents. Thus, the above definition is well defined.

We may easily verify the following statements: for $A, B \subseteq Y$,

(1) $A < B$ if and only if $A <_i B$ for one $i \in Y$.

(2) $A <_i B$ and $A <_j C$ with $i < j$ implies $C <_i B$.

(3) $i \notin A$ implies $A < A \oplus i$.

(4) $A <_i B$ and B an g-interesting intent (i.e. $B = g(B) = B''$) implies $A \oplus i \subseteq B$.

(5) $A <_i B$ and B an g-interesting intent implies $A <_i A \oplus i$.

Analogously to [12, p.67, Theorem 5], we obtain that for a given set $A \subset Y$, if $((A \cap \{1, 2, \ldots, i-1\}) \cup \{i\})''$ is a g-interesting intent, then the smallest g-interesting intent larger than A (with respect to the lectic order) is $A \oplus i$, i being the largest element of Y with $A <_i A \oplus i$.

With the assistance of an EM operator g, we may compute all the g-interesting concepts $\mathcal{B}_g(X, Y, I) = \{(P, Q) \in \mathcal{B}(X, Y, I) \mid Q = g(Q) = Q''\}$.

Let S be an EM operator on $Y = \{1, 2, \ldots, n\}$ with $n < \infty$. From Theorem 1 and the above discussion for lectical order, we understand how to search out the S-intents which we are looking for. We summarize this searching process as the following algorithm.

Algorithm 1 For generating all the S-intents of a given context (X, Y, I):

Step 1. For $A \subset Y$, we find the lectically next intent by checking all the elements i of $Y \setminus A$ with NextClosure algorithm. It is $((A \cap \{1, 2, \ldots, i-1\}) \cup \{i\})''$ as the "next" intent.

Step 2. If $((A \cap \{1, 2, \ldots, i-1\}) \cup \{i\})''$ is an S-intent, then $A \oplus i$ is the intent that we have been looking for.

Otherwise, let $A := ((A \cap \{1, 2, \ldots, i-1\}) \cup \{i\})''$ and go to Step 1.

Since Y is an S-intent and finite, the above algorithm must stop after finite steps. In addition, by NextClosure algorithm, every intent is found by the idea of Step 1. Therefore, all the S-intents may be checked by Step 2. In other words, all the S-intents of (X, Y, I) will be looked for by the above algorithm. For simplicity, we call Algorithm 1 as *NextEM algorithm*.

Remark 3 Using NextEM algorithm to compute the fixed points of $\mathcal{S}$, works with polynomial time delay provided that $S(B)\,(B \subseteq Y)$ can be computed with a polynomial time complexity. Indeed, for each $B \subseteq Y$, B'' can be computed with a polynomial time delay (well-known fact). Thus, if $S(B)$ can be computed in a polynomial time, NextEM can use $\mathcal{S}$ with a polynomial time delay. Still, the number of S-concepts is usually much smaller than the number of all concepts, thus, NextEM with $\mathcal{S}$ is in most situations considerably faster than NextEM with $''$.

In fact, analyzing with Ganter's NextClosure algorithm, we may be assured that to search all the S-intents with the above lectic order, the important is to find

the smallest S-intent in lectic order. Thus, we analyze with S-intents and obtain the following viewpoints.

(3.1.1) Let S be an EM operator defined on Y. We define an operator $f : 2^Y \to 2^Y$ as: for any $A \subseteq E$,

if $\emptyset'' \subset A$, then $f(A) = S(A)$;

if $A = \emptyset''$, then $f(A) = \emptyset''$;

if $A \subset \emptyset''$, then $f(A) = A$.

We may easily decide that f is an EM operator. Additionally, in virtue of Definition 1 and Lemma 1, $\emptyset''$ is the smallest element in $Int_f(X, Y, I)$. Thereby,

if $A \subset \emptyset''$, then A is not an intent;

if $\emptyset'' \subset A$ holds and A is an S-intent, then A is an f-intent and $f(A) = S(A) = A = A''$.

Furthermore, if $A \subset E$ holds and A is an f-intent, then there is $f(A) = A = A''$. So, we obtain $\emptyset'' \subseteq A'' = A$. Hence, A is also an S-intent.

Considered the definition of f, we may determine that:

if $\emptyset'' \neq S(\emptyset'')$, then $Int_S(X, Y, I) = Int_f(X, Y, I) \setminus \emptyset''$;

if $\emptyset'' = S(\emptyset'')$, then $Int_S(X, Y, I) = Int_f(X, Y, I)$.

Summing up, $A \neq \emptyset''$ is an S-intent if and only if A is an f-intent.

(3.1.2) Considered the lectic order $<_i$, we may assert $S(\emptyset'') <_i S(A'')$ $(A \subseteq Y)$. Moreover, if $S(\emptyset'')$ is an S-intent, then $S(\emptyset'')$ will be the smallest S-intent in lectic order. Therefore, we receive $f(\emptyset'') <_i f(A'') = S(A'')$ for any $A \subseteq E$.

Let S be an EM-operator on $Y = \{1, 2, \ldots, n\}$ with $n < \infty$. Let f be defined from S by means of (3.1.1). According to the discussion in (3.1.1) and (3.1.2), we may search out $Int_S(X, Y, I)$ with the following Algorithm 2.

Before introducing Algorithm 2, we provide an assistant algorithm.

Algorithm 2.1 For generating all f-intents of a given context (X, Y, I):

The lectically smallest f-intent is $\emptyset''$.

For a given set $B \subset Y$, we find out the lectically next f-intent by checking all elements i of $Y \setminus B$, starting from the largest one and continuing in a descending order until for the first time $B <_i B \oplus i$.

Then, $B \oplus i$ is the "next" f-intent we have been looking for, where $B \oplus i := f(((B \cap \{1, 2, \ldots, i-1\}) \cup \{i\})'')$.

Algorithm 2 For generating all the S-intents of a given context (X, Y, I):

Step 1. Using Algorithm 2.1, generate all f-intents of (X, Y, I).

Step 2. Output $Int_f(X, Y, I)$ in lectic order.

Step 3. To determine $\emptyset'' = S(\emptyset'')$ is yes or no.

If yes, then $Int_S(X, Y, I) = Int_f(X, Y, I)$.

If no, then $Int_S(X, Y, I) = Int_f(X, Y, I) \setminus \emptyset''$.

Remark 4 Comparing Algorithm 2.1 with NextClosure algorithm in [12, pp.67-68], we may point out that Algorithm 2.1 is a little similar to NextClosure algorithm. We are well known that NextClosure algorithm completes with polynomial time delay. Hence, we may believe that the two steps of Algorithm 2.1 and Step 2 in Algorithm 2 completes in polynomial time delay if $S(A'')$ is for $A \subseteq Y$. Evidently, Step 3 in Algorithm 2 will be completed with polynomial time delay if $S(\emptyset'')$ is. In one word, Algorithm 2 completes with polynomial time complexity if $S(A'')$ is for any $A \subseteq Y$.

3.2. **Bases of Interesting Attribute Implications.** Let (X, Y, I) be a context. In this section, we may state that all S-concepts can be described by particular sets of "interesting" implications between attributes. We present a way to compute minimal sets of such implications. We suppose that Y is finite.

If we focus only on "interesting models" of sets of attribute implications (or sets of "interesting attribute implications"), we naturally come to the following notions of an S-implication and an S-model.

Definition 4 Let Y be a set of attributes, $S : 2^Y \to 2^Y$ be an EM operator, and T be a set of attribute implications in Y. $M \subseteq Y$ is called an *S-model* of T if M is a set of S-attributes and $M \in Mod(T)$ holds. The system of all S-models of T is denoted by $Mod_S(T)$.

A set T of implications is called *S-complete* in (X, Y, I) if $Mod_S(T) = Int_S(X, Y, I)$. A set T of implications is called an *S-basis* of (X, Y, I) if T is S-complete in (X, Y, I) and no proper subset of T is S-complete in (X, Y, I).

Before finding particular S-bases, we introduce a definition.

Definition 5 Let (X, Y, I) be a formal context, $S : 2^Y \to 2^Y$ be an EM operator, and $\mathcal{S}$ be defined as $(*)$. A set P of S-attributes is called an *S-pseudo-intent* of (X, Y, I) if $P \subset \mathcal{S}(P)$, and for each S-pseudo-intent Q of (X, Y, I) such that $Q \subset P$, we have $\mathcal{S}(Q) \subseteq P$.

In fact, the notion of S-pseudo-intent is the generalization of the notion of a pseudo-intent in [12, p.83] and also the generalization of the notion of a C-pseudo-intent in [3]. The following result will illustrate the role of Definition 5.

Theorem 2 Let (X, Y, I) be a formal context with $|Y| < \infty$, $S : 2^Y \to 2^Y$ be an EM operator and $\mathcal{S}$ be defined as $(*)$. Let

$$T = \{P \Rightarrow \mathcal{S}(P) \mid P \text{ is an } S\text{-pseudo-intent of } (X, Y, I)\}. \qquad (**)$$

Then, T is an S-basis of (X, Y, I).

Proof First, we check that T is S-complete. That is, we may check $Mod_S(T) = Int_S(X, Y, I)$ by showing both inclusions.

"$\subseteq$" Let $M \in Mod_S(T)$.

This means $M \in Mod(T)$ and $M = S(M)$. Hence, M is a set of S-attributes. By contradiction, suppose $M \notin Int_S(X, Y, I)$, i.e. $M \subset \mathcal{S}(M)$, since Theorem 1 and $\mathcal{S}$ satisfies (s1). Now, for each S-pseudo-intent Q, we may obtain $M \models Q \Rightarrow \mathcal{S}(Q)$

in view of $M \in Mod_S(T)$. Thus, if $Q \subset M$, then $\mathcal{S}(Q) \subseteq M$. Therefor, M is an S-pseudo-intent of (X, Y, I). So, we obtain $M \Rightarrow \mathcal{S}(M) \in T$. On the other hand, $M \not\models M \Rightarrow \mathcal{S}(M)$ in virtue of $\mathcal{S}(M) \not\subseteq M$. This is a contradiction to $M \in Mod_S(T)$. Moreover, we attain $M \in Int_S(X, Y, I)$.

"$\supseteq$" Let $M \in Int_S(X, Y, I)$.

Certainly, M is a set of S-attributes in light of $M \in Int_S(X, Y, I)$. Using Theorem 1, we receive $M = \mathcal{S}(M)$. Let P be an S-pseudo-intent of (X, Y, I) and $P \subseteq M$. We easily obtain $\mathcal{S}(P) \subseteq \mathcal{S}(M) = M$ since $\mathcal{S}$ satisfies (s2). Therefore, we receive $M \models P \Rightarrow \mathcal{S}(P)$. Moreover, we attain $M \in Mod_S(T)$.

Second, we check that T is an S-basis.

T is obviously a set of implications. For each S-pseudo-intent P and any S-pseudo-intent Q with $Q \neq P$, if $Q \subset P$, then $\mathcal{S}(Q) \subseteq P$ according to Definition 5. Moreover, $P \models Q \Rightarrow \mathcal{S}(Q)$ holds. Thus, P is an S-model of $T_P = T - \{P \Rightarrow \mathcal{S}(P)\}$ which gives $Mod_S(T_P) \supseteq Int_S(X, Y, I)$. Thereby, T_P is not an S-complete.

With Theorem 2, we may reveal that in order to search out an S-basis of (X, Y, I), it suffices to compute all S-pseudo-intents. Therefor, we now turn our attention to the computation of S-pseudo-intents. Given a set T of AD-formulas define sets $B^{T_i}, \mathfrak{S}_T(B)$ $(i \in \mathbb{N}_0)$:

$$B^{T_i} = \begin{cases} B, & \text{if } i = 0 \\ S(B^{T_{i-1}} \cup \bigcup\{D \mid A \Rightarrow D \in T \text{ and } A \subset B^{T_{i-1}}\}), & \text{if } i \geq 1 \end{cases}$$

$$\mathfrak{S}_T(B) = \bigcup_{i=0}^{\infty} B^{T_i}. \qquad (***)$$

The operator $\mathfrak{S}_T : 2^Y \to 2^Y$ has the following property.

Theorem 3 Let (X, Y, I) be a formal context with $|Y| < \infty$, T be defined as $(**)$, and $\mathcal{P}_S$ be the system of all S-pseudo-intents of (X, Y, I). Then, $\mathfrak{S}_T$ defined by $(***)$ is an operator such that $\{B \subseteq Y \mid \mathfrak{S}_T(B) = B\} = \mathcal{P}_S \cup Int_S(X, Y, I)$.

Proof We check $\{B \subseteq Y \mid \mathfrak{S}_T(B) = B\} = \mathcal{P}_S \cup Int_S(X, Y, I)$ by showing both inclusions.

"$\subseteq$": let $B = \mathfrak{S}_T(B)$.

If $B \notin Int_S(X, Y, I)$, then it suffices to check that B is an S-pseudo-intent. With $|Y| < \infty$, we obtain $B = \mathfrak{S}_T(B) = B^{T_{i_0}}$ for some $i_0 \in \mathbb{N}_0$. That is, B is of the form $S(\cdots)$ and following that B is a set of S-attributes. Moreover, for each S-pseudo-intent Q, if $Q \subset B$, then $\mathcal{S}(Q) \subseteq B$ according to $B = \mathfrak{S}_T(B) = B^{T_{i_0}} = B^{T_{i_0+1}} = S(B \cup \bigcup\{\mathcal{S}(A) \mid A \Rightarrow \mathcal{S}(A) \in T, A \subset B\}) = S(B)$. Therefore, B is an S-pseudo-intent.

"$\supseteq$": Clearly, for each S-intent B, there is $B = \mathcal{S}(B)$ by Theorem 1. Hence, we obtain $B^{T_0} = S(B \cup \bigcup\{\mathcal{S}(A) \mid A \Rightarrow \mathcal{S}(A) \in T, A \subset B\}) = S(B^{T_0}) = S(B)$. Furthermore, we receive $B^{T_i} = B$, $(i \in \mathbb{N}_0)$. So, B is a fixed point of $\mathfrak{S}_T$.

Theorem 3 informs us that the set of all S-pseudo-intents and all S-intents is the set of all fixed points of EM operator $\mathfrak{S}_T$ (obviously, $\mathfrak{S}_T$ is an EM operator). This provides us with a way to determine an S-basis: we may compute all fixed points of $\mathfrak{S}_T$, and this follows that $\{P \mid P = \mathfrak{S}_T(P) \text{ and } P \neq \mathcal{S}(P)\}$ is the system of all S-pseudo-intents. Thus, we may express that $T = \{P \Rightarrow \mathcal{S}(P) \mid P = \mathfrak{S}_T(P) \text{ and } P \neq \mathcal{S}(P)\}$ is an S-basis since Theorem 3. For different, the idea of algorithm is depicted in the following.

Let g be an EM operator on the set $Y = \{1, 2, \ldots, n\}$ of attributes for a context (X, Y, I) with $|Y| < \infty$ and $A \boxplus i = g((A \cap \{1, 2, \ldots, i-1\}) \cup \{i\})$. Similarly to the discussion for NextEM algorithm, we may express that the following statements are true: for $A, B \subseteq Y$,

(1) $A < B$ iff $A <_i B$ for one $i \in Y$.

(2) $A <_i B$ and $A <_j C$ with $i < j$ implies $C <_i B$.

(3) $i \notin A$ implies $A < A \boxplus i$.

(4) $A <_i B$ and B a g-fixed point (i.e. $B = g(B)$) implies $A \boxplus i \subseteq B$.

(5) $A <_i B$ and B a g-fixed point implies $A <_i A \boxplus i$.

Analogously to the proof of [12, p.67, Theorem 5], we obtain that for $A \subset Y$, if $(A \cap \{1, 2, \ldots, i-1\}) \cup \{i\}$ is a g-fixed point, then the smallest fixed point of g larger than A (with respect to the lectic order) is $A \boxplus i$, where i is the largest element of Y with $A <_i A \boxplus i$.

Algorithm 3 For generating all the $\mathfrak{S}_T$-fixed points of a given context (X, Y, I):

Step 0. Let $\{B \subseteq Y \mid B = \mathfrak{S}_T(B)\} = \emptyset$.

Step 1. For a given set $A \subset Y = \{1, 2, \ldots, n\}$, we find the lectically next $\mathfrak{S}_T$-fixed point by checking all elements i of $Y \setminus A$, starting from the largest one and continuing in a descending order until for the first time $A <_i A \boxplus i$.

Step 2. If $A \boxplus i = (A \cap \{1, 2, \ldots, i-1\}) \cup i$, then $\{B \subseteq Y \mid B = \mathfrak{S}_T(B)\} = \{B \subseteq Y \mid B = \mathfrak{S}_T(B)\} \cup (A \boxplus i)$. Otherwise, $A := (A \cap \{1, 2, \ldots, i-1\}) \cup i$ and go to Step 1.

According to $|Y| < \infty$, we may point that after finite steps, we may find out the next $\mathfrak{S}_T$-fixed point containing A with respect to lectic order, and further, all the $\mathfrak{S}_T$-fixed points.

Algorithm 3 is completed with the assistance of Ganter's NextClosure algorithm. We may state that Algorithm 3 completes with polynomial complexity if $S(A)\,(A \subseteq Y)$ is.

Remark 5 Owing to Definition 3, we may indicate that each closure operator $C : 2^Y \to 2^Y$ is an EM operator on Y. Considering with [3], we may believe that all the results in this subsection are the generalization of that in [3]. Furthermore, the results here have much more universal and significant.

4. Conclusions

The main goal of this paper is to emphasize the need for taking into account background knowledge in FCA. A background knowledge represents an additional information regarding the input data that a user may have. Such information can be used in the process of FCA to define what is interesting for the user. In particular, we presented an approach to representation and treatment of background knowledge that concerns user's priorities regarding attributes with an EM operator and their relative importance. This may significantly reduce the number of formal concepts extracted from the input data. We focused on the main notions and presented theoretical foundations, and the ideas of relative algorithms.

Acknowledgement. Granted by NSF of China (61572011) and NSF of Hebei Province (A2017201007).

References

[1] Belohlávek, R., Vychodil V., and Zacpal,J., "Concept lattices constrained by equivalence relations", in: V.Snásel and R.Belohlávek (Eds.): *Proceedings of the CLA 2004 International Workshop on Concept Lattices and their Applications*, Ostrava, September 23-24, 2004, pp. 58-66.

[2] Belohlávek, R., Vychodil, V., and Zacpal, J., "Concept lattices constrained by systems of partitions", *Proceeding Znalosti 2005, 4th Annual Congrence*, Stara Lesna, February 9-11, 2005, pp.5-8.

[3] Belohlávek, R., and Vychodil, V., "Formal concept analysis with constraints by closure operators", in: H.Schärfe, P.Hitzler and P.Øhrstrøm (Eds.), *International Conference on Computational Science 2006*. Berlin: Springer-Verlag, 2006, pp.131-143.

[4] Belohlávek, R., and Vychodil, V., "Semantic entailment of attribute-dependency formulas and their non-redundant bases", in: H.D.Cheng, S.D.Chen and R.Y.Lin (Eds.): *Proceedings of the 9th Joint Conference on Information Sciences, Advances in Intelligent Systems Research*, Kaohsiung, October 8-11, 2006, pp. 747-750.

[5] Belohlávek, R., and Vychodil, V., "Adding background knowledge to formal concept analysis via attribute dependency formulas", in: R.L.Wainwright and H.Haddad (Eds.): *The 23rd Annual ACM Symposium on Applied Computing*, Fortaleza, March 16-20, 2008, pp. 938-943.

[6] Belohlávek, R., and Vychodil, V., "Formal concept analysis with background knowledge: attribute priorities", *IEEE Transactions on Systems, Man and Cybernetics–Part C: Applications and Reviews*, **39:4**(2009), 399-409, .

[7] Boulicaut, J.F., and Jeudy, B., "Constraint-based data mining", in: O.Maimon and L.Rokach. (Eds.), *The Data Mining and Knowledge Discovery Handbook*. New York: Springer, 2005, pp.399-416.

[8] Carpineto, C., and Romano, G.,*Concept Data Analysis. Theory and Applications*. England: J. Wiley & Sons, 2004.

[9] Besson, J., Robardet, C., Boulicaut, J., and Rome, S., " Constraint-based concept mining and its application to microarray data analysis", *Intellligent Data Analysis*, **9:1**(2005), 59-82.

[10] Davey, B.A., and Priestley, H.A., *Introduction to lattices and order*, 2nd edition. Cambridge: Cambridge University Press, 2003.

[11] Davidson, I., and Ravi, S.S., "Hierarchical clustering with constraints: theory and practice", *Lecture Notes in Artificial Intelligence*, **3721**(2005), 59-70.

[12] Ganter, B., and Wille, R., *Formal Concept Analysis: Mathematical Foundations*. New York: Springer-Verlag, 1999.

[13] Ganter,B., Stumme, G., and Wille, R.,(Eds.), *Formal Comcept Analysis: Foundations and Applications*. Berlin: Spring-Verlag, 2005.

[14] Pensa, R.G., Boulicaut, J., Cordero, F., and Atzori, M., "Co-clustering numerical data under user-defined constraints", *Statistical Analysis and Data Mining*, **3:1**(2010), 38-55.

[15] Blachon, S., Pensa, R.G., Besson, J., Robardet, C., Boulicaut, J.F., and Gandrllon, O., "Clustering formal concepts to discover biologically relevant knowledge from gene expression data", *Silico Biology*, **7**(2007), 467-483.

[16] Manning, C.D., Raghavan, P., and Schütze, H., *Introduction to Information Retrieval*. Cambridge: Cambridge University Press, 2008.

[17] Qadi, A.E., Aboutajdine, D., and Ennouary, Y., "Formal concept analysis for information retrieval", *International Journal of Computer Science and Information Security*, **7:2**(2010), 1-7.

[18] Quintero, K., Niel, E., Aguilar, J., and Piétrac, L., "Scheduling operations in a flow network with flexible preventive maintenance: a (max, +) approach", *IAENG Engineering Letters*, **22:1**(2014), 24-33.

[19] Mohaupt, M., and Hilbert, A., "Integration of information systems in cloud computing for establishing a Long-term profitable customer portfolio", *IAENG International Journal of Computer Science*, **40:2**(2013), 124-133.

[20] Muangprathub, J., Boonjing, V., and Pattaraintakorn, P., "A new case-based classification using in cremental concept lattice knowledge", *Data & Knowledge Engineering*, **83**(2013), 39-53.

[21] Alwahaishi, S., Martinovič, J., and Snášel, V., "Publications' classification analysis using concept lattices", *International Journal on New Computer Architectures and their Applications*, **2:1**(2012), 312-323.

[22] Medina, J., "Relating attribute reduction in formal, object-oriented and property-oriented concept lattices", *Computer & Matheamtics with Applications*, **64:6**(2012), 1992-2002.

[23] Belohlávek, R., Baets, B.D., and Konecny, J., "Granularity of attributes in formal concept analysis", *Information Sciences*, **260**(2014), 149-170.

[24] Belohlávek, R., and Vychodil, V., "Closure-based constraints in formal concept analysis", *Discrete Applied Mathematics*, **161:13-14**(2013), 1894-1911.

[25] Belohlávek, R., Kostak, M., and Osicka, P., "Formal concept analysis with background knowledge: a case study in paleobiological taxonomy of belemnites", *International Journal of General Systems*, **42:4**(2013), 426-440.

[26] Mao, H., and Li, B., "Formal concept analysis of attributes constrained by equivalent relation", *Computer Engineering and Applications*, **46:36**(2010), 158-160. (in Chinese with English summary)