


Article

A Fast Algorithm for Updating Negative Concept Lattices with Increasing the Granularity Sizes of Attributes

Junping Xie , Liuhai Zhang and Jing Yang

Faculty of Transportation Engineering, Kunming University of Science and Technology, Kunming 650500, China; 20202106031@stu.kust.edu.cn (L.Z.); 20222106017@stu.kust.edu.cn (J.Y.)

* Correspondence: 20110206@kust.edu.cn

Abstract: In this paper, firstly, we studied the relationship between negative concept lattices with increasing the granularity sizes of the attributes. Aiming to do this, negative concepts and covering relations were both classified into three types, and the sufficient and necessary conditions of distinguishing these kinds of negative concepts and covering relations are given, respectively. Further, based on the above analysis, an algorithm for updating negative concept lattices after the increase is proposed. Finally, the experimental results demonstrated that our algorithm performed significantly better than the direct construction algorithm.

Keywords: concept analysis; negative concept lattices; update of negative concept lattices; increase of the granularity sizes of attributes

MSC: 06C15; 68T30

Citation: Xie, J.; Zhang, L.; Yang, J. A Fast Algorithm for Updating Negative Concept Lattices with Increasing the Granularity Sizes of Attributes. *Mathematics* **2023**, *11*, 3229. <https://doi.org/10.3390/math11143229>

Academic Editor: Theodore E. Simos

Received: 11 June 2023

Revised: 18 July 2023

Accepted: 19 July 2023

Published: 22 July 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Formal concept analysis (FCA), proposed by Wille [1], has been widely applied in knowledge discovery [2–6]. In FCA, formal contexts, formal concepts, and concept lattices are the three cornerstones [7]. To deal with various data, formal contexts are extended to fuzzy contexts [8–16], decision contexts, incomplete contexts [17–20], multi-scale contexts [21–24], and triadic contexts [25]. Most of the studies on FCA concentrate on the following topics: concept lattices' construction [26–28], knowledge reduction [29,30], rule acquisition [19,31–36], three-way FCA [37–44], and concept learning [45,46].

In classic FCA, attention is only paid to positive attributes, while negative attributes are neglected. In fact, positive attributes and negative attributes are of equal importance in many fields. Qi et al. [37] proposed negative operators. To be specific, using negative operators, the attributes that are not owned by any given set of objects and the objects that do not have any given set of attributes can be obtained. Based on negative operators, negative concepts and negative concept lattices are defined.

In classic contexts, an attribute is either owned or not owned by an object, that is the relation between an object and an attribute is a binary relation. However, in real data, attributes could be many-valued, and these many-valued attributes are called general attributes. A data table with general attributes needs to be transformed into a one-valued context by the scaling approach [7]. The scaling approach is to replace each general attribute with a sequence of one-valued attributes considered as the corresponding values of the general attribute at certain granularity sizes. For a general attribute, finding appropriate granularity sizes is expected according to a specific requirement. Since the requirements are different, changing the granularity sizes of general attributes is common.

If a classic or negative concept lattice is reconstructed every time whenever the granularity sizes of the attributes change, it is obvious that it will be very computationally expensive and time consuming. In order to avoid reconstructing classic or negative concept lattices, studying the issues of updating classical or negative concept lattices is important when

the granularity sizes of attributes change. Many researchers [8,47–49] have investigated how to update classic concept lattices when the granularity sizes of attributes change. However, little work has been performed on how to update negative concept lattices when the granularity sizes of attributes change. This paper focused on how to update negative concept lattices with increasing the granularity sizes of the attributes.

The rest of this paper is structured as follows. Section 2 recalls the basic knowledge about formal concept analysis. Section 3 studies the relationship between negative concept lattices (negative concepts and covering relations) before and after increasing the granularity sizes of attributes. Section 4 develops a new algorithm (named NCL-Fold) to update the negative concept lattices after the increase. Section 5 carries out the comparison experiment for verifying the effectiveness of our algorithm. Section 6 concludes this paper.

2. Preliminaries

In this section, we present the related notions and propositions in FCA. A detailed description of them can be found in [1,37,47]. We write $\mathcal{P}(\cdot)$ to denote the power set of a set.

A formal context is a triple (G, M, I) , where G refers to a set of objects, M refers to a set of attributes, and I refers to a binary relation between G and M . Here, $(x, m) \in I$ (or $(x, m) \notin I^c$) denotes that the object x has the attribute m , while $(x, m) \notin I$ (or $(x, m) \in I^c$) denotes that the object x does not have the attribute m , where $I^c = (G \times M) - I$.

Let (G, M, I) be a formal context. For $X, Y \subseteq G$ and $A, B \subseteq M$, the positive derivative operator $*$ and the negative derivative operator $\bar{*}$ are as follows:

$$\begin{aligned} * : \mathcal{P}(G) &\rightarrow \mathcal{P}(M), X^* = \{m \in M \mid \forall x \in X, (x, m) \in I\}, \\ * : \mathcal{P}(M) &\rightarrow \mathcal{P}(G), B^* = \{x \in G \mid \forall m \in B, (x, m) \in I\}, \\ \bar{*} : \mathcal{P}(G) &\rightarrow \mathcal{P}(M), X^{\bar{*}} = \{m \in M \mid \forall x \in X, (x, m) \in I^c\}, \\ \bar{*} : \mathcal{P}(M) &\rightarrow \mathcal{P}(G), B^{\bar{*}} = \{x \in G \mid \forall m \in B, (x, m) \in I^c\}. \end{aligned}$$

For these operators, there exist some useful properties. Take the negative derivative operator for example. For $X, X_1, X_2 \subseteq G$ and $A, A_1, A_2 \subseteq M$, then:

- (1) $X_1 \subseteq X_2 \Rightarrow X_2^{\bar{*}} \subseteq X_1^{\bar{*}}$ and $A_1 \subseteq A_2 \Rightarrow A_2^{\bar{*}} \subseteq A_1^{\bar{*}}$;
- (2) $X \subseteq X^{\bar{*}\bar{*}}$ and $A \subseteq A^{\bar{*}\bar{*}}$;
- (3) $X^{\bar{*}} = X^{\bar{*}\bar{*}\bar{*}}$ and $A^{\bar{*}} = A^{\bar{*}\bar{*}\bar{*}}$;
- (4) $X \subseteq A^{\bar{*}} \Leftrightarrow A \subseteq X^{\bar{*}}$;
- (5) $(X_1 \cup X_2)^{\bar{*}} = X_1^{\bar{*}} \cap X_2^{\bar{*}}$ and $(A_1 \cup A_2)^{\bar{*}} = A_1^{\bar{*}} \cap A_2^{\bar{*}}$;
- (6) $(X_1 \cap X_2)^{\bar{*}} \supseteq X_1^{\bar{*}} \cup X_2^{\bar{*}}$ and $(A_1 \cap A_2)^{\bar{*}} \supseteq A_1^{\bar{*}} \cup A_2^{\bar{*}}$.

Using these derivative operators, formal concepts and negative concepts are formed. For $X, Y \subseteq G$ and $A, B \subseteq M$:

- (1) A formal concept is a pair (X, A) , iff $X^* = A$ and $A^* = X$;
- (2) A negative concept (N-concept) is a pair (Y, B) , iff $Y^{\bar{*}} = B$ and $B^{\bar{*}} = Y$.

The partial-order relationship among formal concepts and N-concepts is defined as follows, respectively. For formal concepts (X_1, A_1) and (X_2, A_2) , N-concepts (Y_1, B_1) and (Y_2, B_2) of the formal context (G, M, I) :

$$\begin{aligned} (X_1, A_1) \leq (X_2, A_2) &\Leftrightarrow X_1 \subseteq X_2 \Leftrightarrow A_2 \subseteq A_1; \\ (Y_1, B_1) \leq (Y_2, B_2) &\Leftrightarrow Y_1 \subseteq Y_2 \Leftrightarrow B_2 \subseteq B_1. \end{aligned}$$

All formal concepts and N-concepts generated from the formal context (G, M, I) compose the formal concept lattice and the N-concept lattice under the above partial-order relationships, respectively. The formal concept lattice and N-concept lattice of the formal context (G, M, I) are denoted by $FCL(G, M, I)$ and $NCL(G, M, I)$, respectively.

On the basis of the partial-order \leq , the definition of the covering relation \prec is proposed. For formal concepts (X_1, A_1) and (X_2, A_2) , N-concepts (Y_1, B_1) and (Y_2, B_2) of the formal context (G, M, I) :

$$(X_1, A_1) \prec (X_2, A_2) \Leftrightarrow (X_1, A_1) \leq (X_2, A_2) \text{ and } \{(X_3, A_3) \in FCL(G, M, I) | (X_1, A_1) < (X_3, A_3) < (X_2, A_2)\} = \emptyset;$$

$$(Y_1, B_1) \prec (Y_2, B_2) \Leftrightarrow (Y_1, B_1) \leq (Y_2, B_2) \text{ and } \{(Y_3, B_3) \in NCL(G, M, I) | (Y_1, B_1) < (Y_3, B_3) < (Y_2, B_2)\} = \emptyset.$$

On the basis of the partial-order \prec , the definitions of lower neighbors (or children) and upper neighbors (or parents) are proposed. For formal concepts (X_1, A_1) and (X_2, A_2) , N-concepts (Y_1, B_1) and (Y_2, B_2) of the formal context (G, M, I) :

(1) If $(X_1, A_1) \prec (X_2, A_2)$, then (X_1, A_1) is called a lower neighbor (or a child) of (X_2, A_2) or (X_2, A_2) is called an upper neighbor (or a parent) of (X_1, A_1) ;

(2) If $(Y_1, B_1) \prec (Y_2, B_2)$, then (Y_1, B_1) is called a lower neighbor (or a child) of (Y_2, B_2) or (Y_2, B_2) is called an upper neighbor (or a parent) of (Y_1, B_1) .

In addition, the notions of a granularity tree, cuts, and increasing the granularity sizes are given. A granularity tree (g-tree) of attribute m is a rooted tree, in which each node of the tree is labeled as a unique attribute name, and if, for any node v , the children of node v are nodes v_1, v_2, \dots, v_n , then $\{v_1^*, v_2^*, \dots, v_n^*\}$ must be a partition of v^* . For a set of nodes \mathcal{C} in the g-tree of attribute m , if, for each leaf node v_0 , there is only one node $v \in \mathcal{C}$ on the path from the root m to v_0 , the set of nodes \mathcal{C} is a cut at certain granularity sizes in the g-tree. For two cuts in a given g-tree \mathcal{C}^1 and \mathcal{C}^2 , if, for any $v \in \mathcal{C}^1$, there exists $v' \in \mathcal{C}^2$ such that $v^* \subseteq v'^*$, \mathcal{C}^1 is called a refinement of \mathcal{C}^2 , denoted by $\mathcal{C}^1 \leq \mathcal{C}^2$. Increasing the granularity sizes of an attribute m replaces the existing finer cut $H^f = \{v_i | i = 1, \dots, n^f\}$ ($n^f \geq 2$) of the attribute m with another coarser cut $H^c = \{v'_j | j = 1, \dots, n^c\}$ ($n^c \geq 2$) of the attribute m , where H^f and H^c are two different cuts in the g-tree of attribute m , and $H^f \leq H^c$.

Example 1. Table 1 depicts a context $T_1 = (U, M)$, where a, b , and c are three one-valued attributes and y is a many-valued attribute. The g-tree for attribute y is displayed in Figure 1. In the g-tree, there are five cuts: $\{y\}$, $\{y'_1, y'_2\}$, $\{y'_1, y_3, y_4\}$, $\{y_1, y_2, y'_2\}$, and $\{y_1, y_2, y_3, y_4\}$. In Table 1, the cut $\{y_1, y_2, y_3, y_4\}$ is used. In Table 1, for attribute y , by replacing the finer cut $\{y_1, y_2, y_3, y_4\}$ with the coarser cut $\{y'_1, y'_2\}$, which increases the granularity sizes, the context T_1 is transformed into the context T_2 (i.e., Table 2).

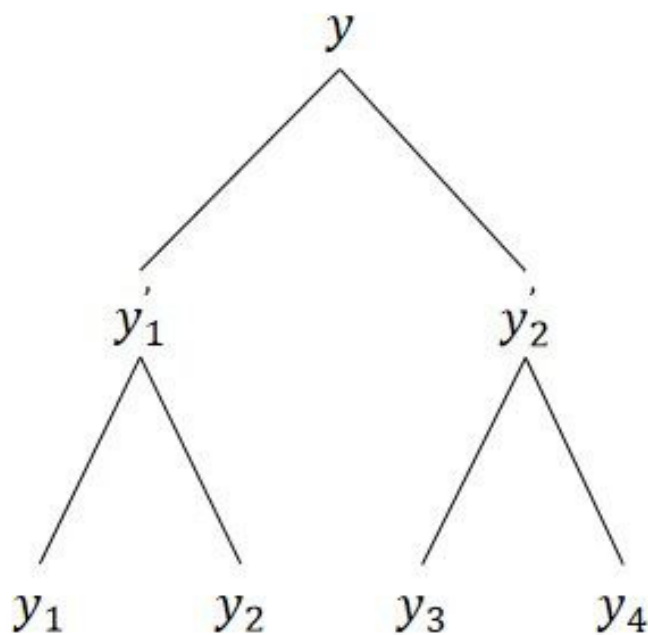


Figure 1. The g-tree for attribute y in Table 1.

Table 1. The context T_1 .

U	a	b	c	y
x_1	1	0	1	y_1
x_2	0	1	0	y_2
x_3	0	0	1	y_3
x_4	1	0	1	y_4

Table 2. The context T_2 .

U	a	b	c	y
x_1	1	0	1	y'_1
x_2	0	1	0	y'_1
x_3	0	0	1	y'_2
x_4	1	0	1	y'_2

For a formal context (G, M, I) , before and after increasing the granularity sizes of the attributes, the negative derivative operators, the N-concept lattices, the subconcept–supconcept relations, and the covering relations are denoted by $\overline{\ast}^f, NCL^f, \leqslant_f, \prec_f$, and $\overline{\ast}^c, NCL^c, \leqslant_c, \prec_c$, respectively, hereafter.

For the finer cut H^f and the coarser cut H^c of attribute m , there exists the attribute-value-transformation function $\alpha : H^f \rightarrow H^c$ such that $\alpha(v_i) = v'_j$. On the basis of the function $\alpha : H^f \rightarrow H^c$, the following mappings are defined:

- $\beta : \mathcal{P}(H^f) \rightarrow \mathcal{P}(H^c)$ such that $\beta(V) = \{\alpha(v) \in H^c | v \in V\}$;
- $\gamma : H^c \rightarrow \mathcal{P}(H^f)$ such that $\gamma(v'_j) = \{v \in H^f | \alpha(v) = v'_j\}$;
- $\delta : H^f \rightarrow \mathcal{P}(H^f)$ such that $\delta(v) = \gamma(\alpha(v))$;
- $\lambda : \mathcal{P}(H^f) \rightarrow \mathcal{P}(H^f)$ such that $\lambda(V) = \{v \in H^f | \delta(v) \subseteq V, v \in V\}$;
- $\mu : \mathcal{P}(H^f) \rightarrow \mathcal{P}(H^c)$ such that $\mu(V) = \{\delta(v) \in H^c | \delta(v) \subseteq V, v \in V\}$.

3. Relationship between N-Concept Lattices before and after Increasing the Granularity Sizes of Attributes

In this section, we discuss the relationship between N-concept lattices before and after increasing the granularity sizes of the attributes. The following theorems are useful to discuss the issue.

Theorem 1. Let NCL^f and NCL^c be the N-concept lattices of a given formal context before and after increasing the granularity sizes of an attribute m , respectively, and $H^f = \{v_i | i = 1, \dots, n^f\}$ and $H^c = \{v'_j | j = 1, \dots, n^c\}$ be the cuts of the attribute m before and after the increase. If $(X, A) \in NCL^f$, then, for each $v' \in H^c$, $X \cap v'^{\overline{\ast}^c}$ is an extent in NCL^c .

Proof. It is obvious that increasing the granularity sizes of an attribute m is equal to adding attributes in H^c and removing attributes in H^f . In addition, for $v' \in H^c$, we have $v'^{\overline{\ast}^c} = \bigcap_{\alpha(v)=v'} v^{\overline{\ast}^f}$. Thus, for each $v' \in H^c$, $X \cap v'^{\overline{\ast}^c}$ must be an extent in NCL^c if X is an extent in NCL^f . □

Theorem 2. Let NCL^f and NCL^c be the N-concept lattices of a given formal context before and after increasing the granularity sizes of an attribute m , respectively. If (X, A) is an N-concept in NCL^c , then X is an extent in NCL^f .

Proof. We discuss two situations for any $(X, A) \in NCL^c$:

- (i) Assume $A \cap H^c = \emptyset$. Obviously, $A^{\overline{\ast}^c} = A^{\overline{\ast}^f} = X$ holds. Thus, we can obtain that X is an extent in NCL^f .

(ii) Assume $A \cap H^c \neq \emptyset$. Since $v^{\overline{\text{sc}}} = \bigcap_{a(v)=v'} v^{\overline{\text{sf}}}$ holds for $v' \in H^c$, we can obtain $(A \cap H^c)^{\overline{\text{sc}}} = \Theta^{\overline{\text{sf}}}$, where $\Theta = \bigcup_{v' \in A \cap H^c} \gamma(v')$. Hence, $A^{\overline{\text{sc}}} = (A - H^c)^{\overline{\text{sc}}} \cap (A \cap H^c)^{\overline{\text{sc}}} = (A - H^c)^{\overline{\text{sf}}} \cap \Theta^{\overline{\text{sf}}} = ((A - H^c) \cup \Theta)^{\overline{\text{sf}}} = X$. Consequently, X is an extent in NCL^f .
 Finally, we can conclude that X is an extent in NCL^f for any $(X, A) \in NCL^c$.
 \square

Based on Theorem 2, we can easily conclude the following theorem.

Theorem 3. Let NCL^f and NCL^c be the N -concept lattices of a given formal context before and after increasing the granularity size of an attribute m , respectively. There does not exist new N -concepts in NCL^c .

3.1. Relationship between N -Concepts before and after Increasing the Granularity Sizes of Attributes

We can describe N -concepts in N -concept lattices before and after increasing the granularity sizes of attributes in terms of the following definition.

Definition 1. Let NCL^f and NCL^c be the N -concept lattices of a given formal context before and after increasing the granularity sizes of an attribute m , respectively, and $H^f = \{v_i | i = 1, \dots, n^f\}$ and $H^c = \{v'_j | j = 1, \dots, n^c\}$ be the cuts of the attribute m before and after the increase, respectively. Then:

- (1) If $(X, A) \in NCL^f$ and $(X, A) \in NCL^c$, (X, A) is an old N -concept, denoted by $class(X, A) = \text{“old”}$;
- (2) If $(X, A) \in NCL^f$ and X is not an extent of any N -concept in NCL^c , (X, A) is a deleted N -concept, denoted by $class(X, A) = \text{“deleted”}$;
- (3) if $(X, A) \in NCL^f$ with $A \cap H^f \neq \emptyset$ and $(X, (A - H^f) \cup \mu(A \cap H^f)) \in NCL^c$, (X, A) is a tight N -concept, denoted by $class(X, A) = \text{“tight”}$.

Next, we give the necessary and sufficient conditions of each category.

Theorem 4. For an N -concept $(X, A) \in NCL^f$, we have:

- (1) (X, A) is an old N -concept if and only if $A \cap H^f = \emptyset$;
- (2) (X, A) is a deleted N -concept if and only if the following statements are true: (i) $A \cap H^f \neq \emptyset$; (ii) there exists at least one N -concept (Y, C) among the parents of (X, A) in NCL^f such that (Y, C) satisfies the two conditions: $C - H^f = A - H^f$ and $\lambda(C \cap H^f) = \lambda(A \cap H^f)$;
- (3) (X, A) is a tight N -concept if and only if the following statements are true: (i) $A \cap H^f \neq \emptyset$; (ii) there does not exist such an N -concept (Y, C) among the parents of (X, A) in NCL^f such that (Y, C) satisfies the two conditions: $C - H^f = A - H^f$ and $\lambda(C \cap H^f) = \lambda(A \cap H^f)$.

Proof.

- (1) (\Rightarrow) If (X, A) is an old N -concept, it is obvious that (X, A) is an N -concept in NCL^f and an N -concept in NCL^c . Next, we prove $A \cap H^f \neq \emptyset$ by using reduction to absurdity. Assume that $A \cap H^f \neq \emptyset$. According to the process of the proof in Theorem 2, we can obtain $A^{\overline{\text{sc}}} = ((A - H^c) \cup \Theta)^{\overline{\text{sf}}} = X$, where $\Theta = \bigcup_{v' \in A \cap H^c} \gamma(v')$.

Hence, $\Theta \subseteq X^{\overline{\text{sf}}} = A$, which contradicts $\Theta \not\subseteq A$. Hence, $A \cap H^f = \emptyset$.

(\Leftarrow) If (X, A) with $A \cap H^f = \emptyset$ is an N -concept in NCL^f , $A^{\overline{\text{sc}}} = A^{\overline{\text{c}}} = X$ and $X^{\overline{\text{sc}}} = X^{\overline{\text{sf}}} = A$ hold. Hence, (X, A) is an N -concept in NCL^f and an N -concept in NCL^c . That is to say, (X, A) is an old N -concept.

- (2) (\Rightarrow) Firstly, by using reduction to absurdity, we prove that $A \cap H^f \neq \emptyset$. Assume that $A \cap H^f = \emptyset$. Because the objects do not change before and after the increase, we can

obtain $X = A^{\overline{f}} = A^{\overline{c}}$. Furthermore, it follows that X is an extent in NCL^c , which is not consistent with $class(X, A) = \text{“deleted”}$. Hence, $A \cap H^f \neq \emptyset$.

Secondly, we prove the rest.

Because of $A \cap H^f \neq \emptyset$, we have $X \subseteq (A \cap H^f)^{\overline{f}} \subseteq \lambda(A \cap H^f)^{\overline{f}}$ and $X \not\subseteq (H^f - A \cap H^f)^{\overline{f}}$. Since $v^{\overline{c}} = \bigcap_{\alpha(v)=v'} v^{\overline{f}}$ holds for $v' \in H^c$, we can obtain: $\mu(A \cap H^f)^{\overline{c}} = \lambda(A \cap H^f)^{\overline{f}}$; $X \subseteq v^{\overline{c}}$ holds when $v' \in \mu(A \cap H^f)$; $X \not\subseteq v^{\overline{c}}$ holds when $v' \notin \mu(A \cap H^f)$. Noting that (X, A) is a deleted N-concept, it follows that $X \subset X^{\overline{c}\overline{c}}$ and $X^{\overline{f}} = (A - H^f) \cup (A \cap H^f)$. Hence, $X \subset X^{\overline{c}\overline{c}} = ((A - H^f) \cup \mu(A \cap H^f))^{\overline{c}} = ((A - H^f) \cup \lambda(A \cap H^f))^{\overline{f}}$. Therefore, there exists an N-concept $(Z, E) \in NCL^f$ such that $(Z, E) = (((A - H^f) \cup \lambda(A \cap H^f))^{\overline{f}})^{\overline{f}}$, $((A - H^f) \cup \lambda(A \cap H^f))^{\overline{f}\overline{f}}$, and $(X, A) \leq_f (Z, E)$.

Then, we prove that E satisfies the following conditions. Because of $(A - H^f) \cup \lambda(A \cap H^f) \subseteq ((A - H^f) \cup \lambda(A \cap H^f))^{\overline{f}\overline{f}} = (E - H^f) \cup (E \cap H^f)$ and $(E - H^f) \cup (E \cap H^f) = E \subseteq A = (A - H^f) \cup (A \cap H^f)$, we have $A - H^f \subseteq E - H^f \subseteq A - H^f$ and $\lambda(A \cap H^f) \subseteq \lambda(E \cap H^f) \subseteq \lambda(A \cap H^f)$. That is to say, $E - H^f = A - H^f$ and $\lambda(E \cap H^f) = \lambda(A \cap H^f)$.

Finally, there must exist an N-concept $(Y, C) \in NCL^f$ such that $(X, A) \prec_f (Y, C)$, $C - H^f = A - H^f$ and $\lambda(C \cap H^f) = \lambda(A \cap H^f)$.

(\Leftarrow) Assume that $A \cap H^f \neq \emptyset$ and there exists one N-concept $(Y, C) \in NCL^f$ with $(X, A) \prec_f (Y, C)$ such that $C - H^f = A - H^f$ and $\lambda(C \cap H^f) = \lambda(A \cap H^f)$. Obviously, $Y \subseteq (A - H^f)^{\overline{f}} \cap \lambda(A \cap H^f)^{\overline{f}}$ holds. By the analysis in (\Rightarrow), it follows that $X^{\overline{c}\overline{c}} = ((A - H^f) \cup \lambda(A \cap H^f))^{\overline{f}}$. Since (Y, C) is a parent of (X, A) and $Y \subseteq (A - H^f)^{\overline{f}} \cap \lambda(A \cap H^f)^{\overline{f}}$ holds, we can conclude that $X \subset Y \subseteq (A - H^f)^{\overline{f}} \cap \lambda(A \cap H^f)^{\overline{f}} = X^{\overline{c}\overline{c}}$. Thus, X is not an extent in NCL^f . That is to say, (X, A) is a deleted N-concept:

- (3) It is easy to reach this conclusion according to (1) and (2) in Theorem 4. □

Based on Theorem 4, we show the following definition and remark.

Definition 2. Let NCL^f be the N-concept lattices of a given formal context before increasing the granularity sizes of an attribute m and $H^f = \{v_i | i = 1, \dots, n^f\}$ be the cuts of the attribute m before the increase. For two N-concepts $(X, A) \in NCL^f$ with $class(X, A) = \text{“deleted”}$ and $(Y, C) \in NCL^f$ with $C - H^f = A - H^f$ and $\lambda(C \cap H^f) = \lambda(A \cap H^f)$:

- (1) (Y, C) is described as a destroyer of (X, A) , and (X, A) is described as a victim of (Y, C) ;
- (2) The N-concept $(Z, E) \in NCL^f$ with $Z = ((A - H^f) \cup \lambda(A \cap H^f))^{\overline{f}}$ and $E = ((A - H^f) \cup \lambda(A \cap H^f))^{\overline{f}\overline{f}}$ is described as a terminator of (X, A) .

Remark 1. For two N-concepts (X, A) and (Y, C) in NCL^f , if (X, A) is a deleted N-concept and (Y, C) is a destroyer of (X, A) , we have:

- (1) The number of destroyers of (X, A) is either equivalent to one or more than one, and the set of all the destroyers is denoted by $Destroyers(X, A)$;
- (2) The number of casualties of (Y, C) is either equivalent to one or more than one, and the set of all the victims of (Y, C) is denoted by $Victims(Y, C)$.
- (3) The number of terminators of (X, A) is equivalent to one, and the only terminator of (X, A) is denoted by $terminator(X, A)$;
- (4) The terminator of (X, A) is the maximum one among all the destroyers of (X, A) .
Combining Definition 2 with Theorem 4 and Remark 1, we have the following theorem.

Theorem 5. For a deleted N-concept (X, A) , we have:

- (1) If (Z, E) is the terminator of (X, A) , then (Z, E) must not be a deleted N-concept;
- (2) If (Y, C) is a destroyer of (X, A) and (Y, C) is not the terminator of (X, A) , then (Y, C) must be a deleted N-concept.

- (3) If (Y, C) is a destroyer of (X, A) and (Y, C) is not a deleted N-concept, then (Y, C) must be the terminator of (X, A) .

Proof.

- (1) Since (Z, E) is the terminator of (X, A) , we can obtain $(Z, E) = (((A - H^f) \cup \lambda(A \cap H^f))^{\bar{f}^f}, ((A - H^f) \cup \lambda(A \cap H^f))^{\bar{f}^f \bar{f}^f}) = (((A - H^f) \cup \mu(A \cap H^f))^{\bar{c}}, ((A - H^f) \cup \mu(A \cap H^f))^{\bar{c} \bar{c}^c})$. Obviously, (Z, E) must be an N-concept in NCL^c . That is to say, (Z, E) is not a deleted N-concept.
- (2) Let (Z, E) be the terminator of (X, A) . Obviously, $(Z, E) = (((A - H^f) \cup \lambda(A \cap H^f))^{\bar{f}^f}, ((A - H^f) \cup \lambda(A \cap H^f))^{\bar{f}^f \bar{f}^f})$. Since (Z, E) is the maximum one among all the destroyers of (X, A) and $(Y, C) \neq (Z, E)$ is a destroyer of (X, A) , we have $E - H^f = C - H^f = A - H^f$ and $\lambda(E \cap H^f) = \lambda(C \cap H^f) = \lambda(A \cap H^f)$. Hence, (Y, C) must be a deleted N-concept and (Z, E) is the terminator of (Y, C) .
- (3) It is easy to reach this conclusion according to (2) in Theorem 5.

□

3.2. Relationship between the Covering Relations before and after Increasing the Granularity Sizes of Attributes

We can describe the covering relations in N-concept lattices before and after increasing the granularity sizes of the attributes in terms of the following definition.

Definition 3. Let NCL^f and NCL^c be the N-concept lattices of a given formal context before and after increasing the granularity sizes of an attribute m , respectively. Then:

- (1) If $(X, A) \prec_f (Y, C)$ and $\bar{f}c(X, A) \prec_c \bar{f}c(Y, C)$ hold, $(X, A) \prec_f (Y, C)$ (or $\bar{f}c(X, A) \prec_c \bar{f}c(Y, C)$) is an old covering relation;
- (2) If $(X, A) \prec_c (Y, C)$ and $\bar{f}c^{-1}(X, A) \not\prec_f \bar{f}c^{-1}(Y, C)$ hold, $(X, A) \prec_c (Y, C)$ is a new covering relation;
- (3) If $(X, A) \prec_f (Y, C)$ holds and at least one of (X, A) and (Y, C) is a deleted N-concept, $(X, A) \prec_f (Y, C)$ is a deleted covering relation.

For $(X, A) \in NCL^f$ and $(Y, C) \in NCL^f$ with $class(Y, C) \neq$ "deleted", two mappings are given by

$$f_c : NCL^f \rightarrow NCL^c, f_c(X, A) = (X^{\bar{c} \bar{c}^c}, X^{\bar{c}^c}),$$

$$\bar{f}c : NCL^f - \{(O, P) \in NCL^f | class(O, P) = \text{"deleted"}\} \rightarrow NCL^c, \bar{f}c(Y, C) = (Y, Y^{\bar{c}^c}).$$

Obviously, f_c is not a bijection, but $\bar{f}c$ is a bijection.

Let $(X_2, A_2) \in NCL^f$ with $class(X_2, A_2) \neq$ "deleted". If (X_2, A_2) is the terminator of a certain N-concept, then we give the following remarks:

$$CAN^\circ(X_2, A_2) = \{(X, A) \in NCL^f | (X, A) \prec_f (X_2, A_2), class(X, A) \neq \text{"deleted"}\},$$

$$CAN(X_2, A_2) = \{(X, A) \in NCL^f | (X, A) \prec_f (X_2, A_2), class(X, A) \neq \text{"deleted"}\} \cup$$

$$\{(X, A) \in NCL^f | terminator(X', A') = (X_2, A_2), (X, A) \prec_f (X', A'), class(X, A) \neq \text{"deleted"}\},$$

$$CAN'(X_2, A_2) = \{(X, A) \in NCL^f | (X, A) \prec_f (X', A'), (X', A') \in CAN(X_2, A_2), class(X, A) \neq \text{"deleted"}\},$$

$$CAN^*(X_2, A_2) = \{(X, A) \in NCL^f | (X, A) \prec_f (X', A') \prec_f (X_2, A_2), terminator(X', A') \neq (X_2, A_2), class(X, A) \neq \text{"deleted"}\},$$

$$CAN^\diamond(X_2, A_2) = \{(X, A) \in CAN(X_2, A_2) | \forall (X', A') \in CAN(X_2, A_2), (X', A') \leq_f (X, A)\},$$

$$CAN^\blacklozenge(X_2, A_2) = CAN^\diamond(X_2, A_2) - \{(X, A) \in NCL^f | (X, A) \prec_f (X_2, A_2), class(X, A) \neq \text{"deleted"}\}.$$

The covering relations among transformed concepts are listed in the following two theorems.

Theorem 6. For $(X_2, A_2) \in NCL^f$, if (X_2, A_2) is not the terminator of any N-concept, then $\{(Z, E) \in NCL^c | (Z, E) \prec_c \bar{f}c(X_2, A_2)\} = \{(X, A) \in NCL^f | (X, A) \prec_f (X_2, A_2)\}$.

Proof. At first, by using reduction to absurdity, we prove that (X_1, A_1) is not a deleted N-concept if (X_2, A_2) is not the terminator of any N-concept and (X_1, A_1) is a child of (X_2, A_2) in NCL^f . Assume that (X_1, A_1) is a deleted N-concept, which implies $A_1 \cap H^f \neq \emptyset$. Obviously, there must exist an N-concept (X_3, A_3) in NCL^f such that $(X_2, A_2) \prec_f (X_1, A_1)$, and (X_3, A_3) is the terminator of (X_1, A_1) . According to the definition of terminators, we can obtain that $A_1 - H^f = A_3 - H^f$ and $\lambda(A_1 \cap H^f) = \lambda(A_3 \cap H^f)$. Because (X_2, A_2) with $class(X_2, A_2) \neq$ "deleted" is not the terminator of any N-concept and (X_1, A_1) is a child of (X_2, A_2) in NCL^f , which implies $A_2 \cap H^f = \emptyset$, $A_2 \subset A_1$, we have $A_2 \subset A_3$. Hence, $A_2 \subset A_3$, which means $(X_1, A_1) \prec_f (X_3, A_3) \prec_f (X_2, A_2)$. That is to say, (X_1, A_1) cannot be a child of (X_2, A_2) in NCL^f , which is inconsistent with the fact that (X_1, A_1) is a child of (X_2, A_2) in NCL^f . Thus, (X_1, A_1) is not a deleted N-concept. Therefore, $X_1^{\bar{f}c} = X_1$ and $X_3 = X_3^{\bar{f}c}$ hold. In addition, since there do not exist new N-concepts in NCL^c , $\bar{f}c(X_1, A_1)$ is a child of $\bar{f}c(X_2, A_2)$ in NCL^c . In summary, $\{(Z, E) \in NCL^c | (Z, E) \prec_c \bar{f}c(X_2, A_2)\} = \{\bar{f}c(X, A) | (X, A) \prec_f (X_2, A_2)\}$ holds. \square

Theorem 7. For $(X_2, A_2) \in NCL^f$, if (X_2, A_2) is the terminator of a certain N-concept, then $\{(Z, E) \in NCL^c | (Z, E) \prec_c \bar{f}c(X_2, A_2)\} = \{\bar{f}c(X, A) | (X, A) \in CAN^\diamond(X_2, A_2)\}$.

Proof. Since there does not exist new N-concepts in NCL^c , it is obvious that $\{(X, A) \in NCL^c | (X, A) \prec_c \bar{f}c(X_2, A_2)\} \subseteq \{\bar{f}c((X, A) | (X, A) \in CAN^\diamond(X_2, A_2))\}$.

Let (X', A') be an N-concept $CAN'(X_2, A_2)$. It is easily seen that there must exist $(X, A) \in CAN(X_2, A_2)$ such that $(X', A') \prec_f (X, A) \prec_f (X_2, A_2)$, which implies $\bar{f}c(X', A') \prec_c \bar{f}c(X, A) \prec_c \bar{f}c(X_2, A_2)$. Hence, $\bar{f}c(X', A') \not\prec_c \bar{f}c(X_2, A_2)$ holds.

Let (Y, C) be an N-concept $CAN^*(X_2, A_2)$. It is easily seen that there must exist two N-concepts (Y', C') and (Y'', C'') in NCL^f such that $(Y, C) \prec_f (Y', C') \prec_f (Y'', C'') \prec_f (X_2, A_2)$, $terminator(Y'', C'') = (X_2, A_2)$ and $terminator(Y', C') \neq (X_2, A_2)$. Hence, according to the definition of terminators, we have $(Y, C) \prec_f terminator(Y', C') \prec_f terminator(Y'', C'') = (X_2, A_2)$, which implies that $terminator(Y', C')$ is an N-concept in $CAN(X_2, A_2)$ or $CAN'(X_2, A_2)$. That is to say, $(Y, C) \in CAN'(X_2, A_2)$ holds. Thus, we can obtain $\bar{f}c(Y, C) \not\prec_c \bar{f}c(X_2, A_2)$ and $CAN^\diamond(X_2, A_2) = CAN(X_2, A_2) \cup CAN'(X_2, A_2) \cup CAN^*(X_2, A_2)$.

In summary, $\{(X, A) \in NCL^c | (X, A) \prec_c \bar{f}c(X_2, A_2)\} \subseteq \{\bar{f}c((X, A) | (X, A) \in CAN(X_2, A_2))\}$. Finally, we can easily conclude that $\{(Z, E) \in NCL^c | (Z, E) \prec_c \bar{f}c(X_2, A_2)\} = \{\bar{f}c(X, A) | (X, A) \in CAN^\diamond(X_2, A_2)\}$. \square

Theorem 8. Let NCL^f and NCL^c be the N-concept lattices of a given formal context before and after increasing the granularity sizes of an attribute m , respectively. For two non-deleted N-concept (X, A) and (Y, C) in NCL^f , the:

- (1) $(Y, C) \prec_f (X, A)$ is an old covering relation, if and only if one of the following statements is true: (i) (X, A) is not the terminator of any N-concept and (Y, C) is an N-concept in $\{(Y', C') \in NCL^f | (Y', C') \prec_f (X, A)\}$; (ii) (X, A) is the terminator of a certain N-concept and (Y, C) is an N-concept in $\{(Y', C') \in NCL^f | (Y', C') \prec_f (X, A), class(X, A) \neq$ "deleted" $\} \cap CAN^\diamond(X, A)$;
- (2) $\bar{f}c(Y, C) \prec_c \bar{f}c(X, A)$ is a new covering relation, if and only if (X, A) is the terminator of a certain N-concept and (Y, C) is an N-concept in $CAN^\diamond(X, A)$.

Example 2. Let us continue with Example 1. Two formal contexts \mathbb{T}_1 and \mathbb{T}_2 (i.e., Tables 3 and 4) are obtained from T_1 and T_2 by the scaling approach. The concept lattices $NCL(\mathbb{T}_1)$ and $NCL(\mathbb{T}_2)$ of \mathbb{T}_1 and \mathbb{T}_2 are displayed in Figure 2 and Figure 3, respectively. In Figures 2 and 3, old N-concepts, deleted N-concepts, and tight N-concepts are the white, red, and blue nodes, respectively.

In Figures 2 and 3, old covering relations, deleted covering relations, and new covering relations are the black, red, and green lines, respectively.

Table 3. The formal context \mathbb{T}_1 .

U	a	b	c	y_1	y_2	y_3	y_4
x_1	1	0	1	1	0	0	0
x_2	0	1	0	0	1	0	0
x_3	0	0	1	0	0	1	0
x_4	1	0	1	0	0	0	1

Table 4. The formal context \mathbb{T}_2 .

U	a	b	c	y'_1	y'_2
x_1	1	0	1	1	0
x_2	0	1	0	1	0
x_3	0	0	1	0	1
x_4	1	0	1	0	1

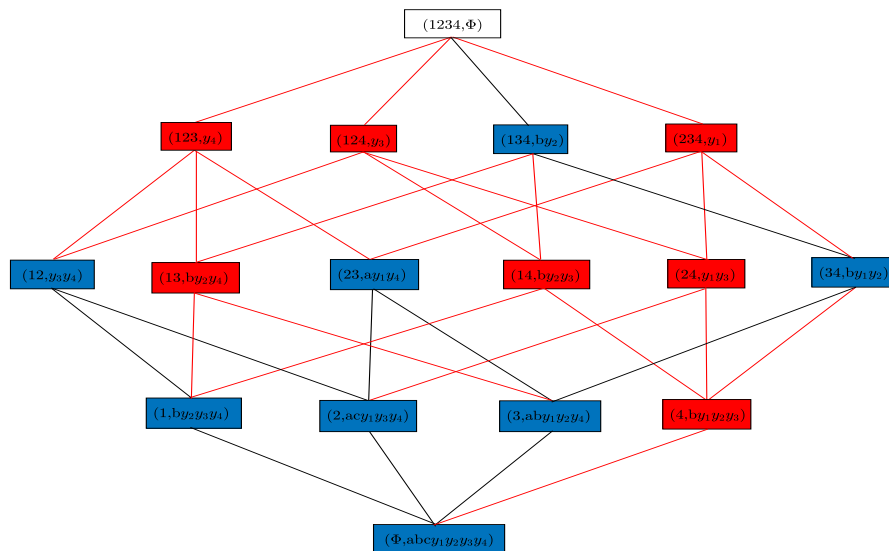


Figure 2. The N-concept lattice of Table 3.

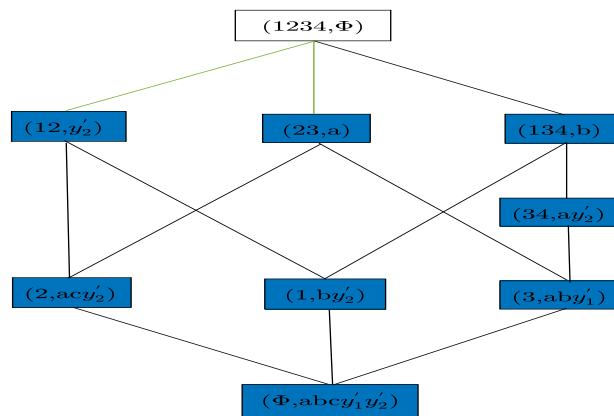


Figure 3. The N-concept lattice of Table 4.

4. The NCL-Fold Algorithm

In this section, we propose a new algorithm (called the NCL-Fold algorithm i.e. Algorithm 1) for increasing the granularity sizes of an attribute m , based on the relationship between N-concept lattices before and after the increase as discussed in Section 3.

Algorithm 1 procedure $NCL - Fold(H^f, H^c, NCL)$.

```

1: Find the top N-concept  $(X_{top}, A_{top})$  in  $NCL$ 
2:  $Process((X_{top}, A_{top}), NCL, H^f, H^c)$ 
3: return  $NCL$ 

```

The procedure $NCL - Fold$ accepts three arguments: the cuts before and after increasing the granularity sizes of attribute m H^f and H^c and the N-concept lattice NCL of a formal context before the increase. $NCL - Fold$ updates the N-concept lattice NCL of a formal context after the increase.

The procedure first finds the top N-concept (X_{top}, A_{top}) in NCL (Line 1). Then, the procedure invokes the following Algorithm 2 to process every N-concept (X, A) in NCL (Line 2) and return updated NCL (Line 3).

Algorithm 2 procedure $Process((X, A), NCL, H^f, H^c)$.

```

1: for each child  $(Y, C)$  of  $(X, A)$ 
2:   if  $(Y, C)$  has not been processed
3:      $Process((Y, C), NCL, H^f, H^c)$ 
4:   end if
5: end for
6: according to Theorem 4 and Definition 1, classify  $(X, A)$  and modify the intent of  $(X, A)$ 
7: if  $(X, A)$  is a deleted N-concept
8:   mark  $(Z, E)$  as a destroyer and  $Victims(Z, E) = Victims(Z, E) \cup \{(X, A)\} \cup Victims(X, A)$ , for every parent  $(Z, E)$  with  $Z - H^f = A - H^f$  and  $\lambda(E \cap H^f) = \lambda(A \cap H^f)$  of  $(X, A)$ 
9: end if
10: mark  $(X, A)$  as a terminator if  $(X, A)$  is not a deleted N-concept and is a destroyer
11: according to Definition 3 and Theorem 8, classify the covering relations relating to  $(X, A)$ , set the new covering relations, and remove the deleted covering relations
12:  $NCL = NCL - Victims(X, A)$ 
13: Mark  $(X, A)$  as processed
14: return  $NCL$ 

```

The procedure $Process$ accepts four arguments: an N-concept (X, A) , NCL , H^f , and H^c . $Process$ traverses all N-concepts in a recursive way.

If a child (Y, C) of (X, A) is not visited, the algorithm recursively invokes $Process$ using the child (Y, C) , NCL , H^f , and H^c as arguments (Lines 1–5). Then, the N-concept (X, A) is classified according to Theorem 4, and the intent of (X, A) is modified according to Definition 1 (Line 6). In addition, for every parent (Z, E) of (X, A) , if (Z, E) satisfies the conditions of a destroyer, (Z, E) is marked as a destroyer and $Victims(Z, E)$ is updated by adding (X, A) and $Victims(X, A)$ to $Victims(Z, E)$ (Lines 7–9). If (X, A) satisfies the conditions of a terminator, (Z, E) is marked as a terminator (Line 10). Next, according to Definition 3 and Theorem 8, the covering relations relating to (X, A) are fixed (Line 11). Of course, $Victims(X, A)$ should be deleted from NCL (Line 12). Finally, (X, A) is marked as processed, and the updated NCL is returned (Lines 13–14).

Now, we analyze the time complexity of Algorithms 1 and 2.

Firstly, we analyze the time complexity of Algorithm 2. The time complexity of Steps 6–10 is $O(max_{parents}|M|^2)$, where $max_{parents}$ is the maximum number of parents of N-concepts in NCL and $|M|$ is the number of attributes. The time complexity of Steps 11–12 is $O(max_{can}^2|M|^2)$, where $max_{can} = \max_{(X,A) \in NCL} \{|Can(X,A)|\}$ and $|M|$ is the number of attributes. Consequently, the time complexity of Algorithm 2 is $O(|NCL|(max_{parents}|M|^2 + max_{can}^2|M|^2))$, where $|NCL|$ is the number of N-concepts in NCL .

Secondly, we can easily see that the time complexity of Algorithm 1 is $O(|NCL|(max_{parents}|M|^2 + max_{can}^2|M|^2))$, as well.

5. Experimental Evaluation

In this section, the main task was to compare our dynamic updating method algorithm NCL-Fold and the traditional method of directly constructing N-concept lattices from datasets using the idea of FastAddIntent [27] in Matlab (Version R2018b). The experimental environment was a server equipped with the 64-bit operating system, Intel(R) Xeon(R) Silver 4210R CPU, 2.40 GHz, 128 GB RAM.

In the experiments, we used contexts that were randomly generated datasets with different fill ratios. The detailed information about the sixteen databases is shown in Table 5. In these datasets, every object owns the same number of attributes according to the fill ratios, and many-valued attribute y owns the same g-tree exhibited in Figure 4. In these datasets, the domain of y is Cut 1 in Figure 4. Figure 5 depicts the running time of our algorithm NCL-Fold and the traditional method on the sixteen random datasets from Cut 1 to Cut 2.

From Figure 5, we can see that: (i) NCL-Fold significantly outperformed the traditional method in every case; (ii) the performance gap was bigger when the number of attributes, fill ratios, and changes of granularity sizes were the same, respectively, and the number of objects increased.

Table 5. The detailed information about the sixteen random datasets.

Database	Objects	Attributes	Fill Ratios
$T_{100 \times 10, 20\%}$	100	9 (Boolean), 1 (discrete, but not Boolean)	20%
$T_{100 \times 10, 30\%}$	100	9 (Boolean), 1 (discrete, but not Boolean)	30%
$T_{100 \times 10, 40\%}$	100	9 (Boolean), 1 (discrete, but not Boolean)	40%
$T_{100 \times 10, 50\%}$	100	9 (Boolean), 1 (discrete, but not Boolean)	50%
$T_{500 \times 10, 20\%}$	500	9 (Boolean), 1 (discrete, but not Boolean)	20%
$T_{500 \times 10, 30\%}$	500	9 (Boolean), 1 (discrete, but not Boolean)	30%
$T_{500 \times 10, 40\%}$	500	9 (Boolean), 1 (discrete, but not Boolean)	40%
$T_{500 \times 10, 50\%}$	500	9 (Boolean), 1 (discrete, but not Boolean)	50%
$T_{1000 \times 10, 20\%}$	1000	9 (Boolean), 1 (discrete, but not Boolean)	20%
$T_{1000 \times 10, 30\%}$	1000	9 (Boolean), 1 (discrete, but not Boolean)	30%
$T_{1000 \times 10, 40\%}$	1000	9 (Boolean), 1 (discrete, but not Boolean)	40%
$T_{1000 \times 10, 50\%}$	1000	9 (Boolean), 1 (discrete, but not Boolean)	50%
$T_{2000 \times 10, 20\%}$	2000	9 (Boolean), 1 (discrete, but not Boolean)	20%
$T_{2000 \times 10, 30\%}$	2000	9 (Boolean), 1 (discrete, but not Boolean)	30%
$T_{2000 \times 10, 40\%}$	2000	9 (Boolean), 1 (discrete, but not Boolean)	40%
$T_{2000 \times 10, 50\%}$	2000	9 (Boolean), 1 (discrete, but not Boolean)	50%

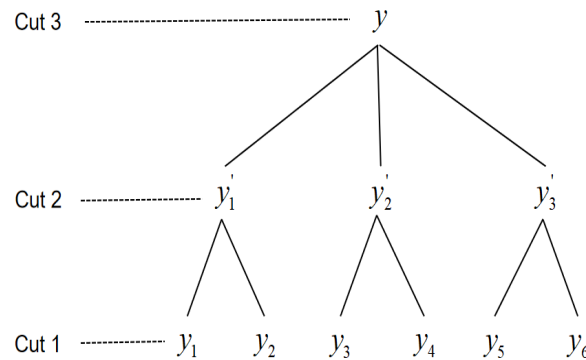


Figure 4. The g-tree for the attribute y .

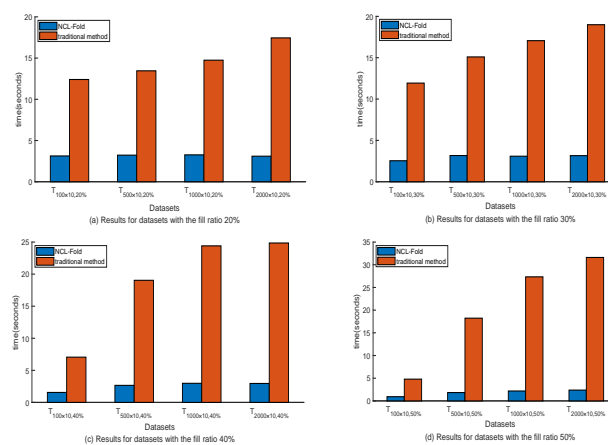


Figure 5. The running time of NCL-Fold and traditional method on Datasets 1–10 from Cut 1 to Cut 2.

6. Conclusions

In this part, we present the main work of this paper and the future research work:

(1) The main work of this paper:

In this paper, we analyzed the relationship between N-concept lattices with increasing the granularity sizes of attributes. Furthermore, we proposed a new algorithm (named NCL-Fold) to update and maintain N-concept lattices with increasing the granularity sizes of attributes. Finally, we conducted experiments, and the experimental results indicated that NCL-Fold has good performance.

(2) Future research work:

Since changing the granularity sizes of attributes includes increasing the granularity sizes of the attributes and decreasing the granularity sizes of the attributes, how to quickly update N-concept lattices when decreasing the granularity sizes of the attributes could be developed in future. In addition, based on our method, how to detect appropriate granularity sizes of the attributes deserves to be studied in the future.

Author Contributions: Conceptualization, J.X.; Methodology, J.X.; Validation, L.Z. and J.Y.; Writing—original draft preparation, J.X.; Writing—review and editing, L.Z. and J.Y.; Visualization, L.Z. and J.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Foundation for Fostering Talents in Kunming University of Science and Technology (No. KKS201902007).

Data Availability Statement: Publicly available datasets are not used in this study, and datasets in this study are randomly generated.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wille, R. Restructuring lattice theory: An approach based on hierarchies of concepts. In *Ordered Sets*; Rival, I., Ed.; Reidel: Dordrecht, The Netherlands; Boston, MA, USA, 1982; pp. 445–470.
2. Aswani Kumar, C.; Srinivas, S. Mining associations in health care data using formal concept analysis and singular value decomposition. *J. Biol. Syst.* **2010**, *18*, 787–807. [[CrossRef](#)]
3. Dias, S.; Vieira, N. Concept lattices reduction: Definition, analysis and classification. *Expert Syst. Appl.* **2015**, *42*, 7084–7097. [[CrossRef](#)]
4. Kuznetsov, S. Machine learning and formal concept analysis. *Lect. Notes Comput. Sci.* **2004**, *2961*, 287–312.
5. Poelmans, J.; Ignatov, D.; Kuznetsov, S.; Dedene, G. Formal concept analysis in knowledge processing: A survey on applications. *Expert Syst. Appl.* **2013**, *40*, 6538–6560. [[CrossRef](#)]
6. Yang, L.; Xu, Y. Decision making with uncertainty information based on lattice-valued fuzzy concept lattice. *J. Univers. Comput. Sci.* **2010**, *16*, 159–177.
7. Ganter, B.; Wille, R. *Formal Concept Analysis: Mathematical Foundations*; Springer: New York, NY, USA, 1999.
8. Bělohlávek, R.; Sklenář, V.; Zackpal, J. Crispily generated fuzzy concepts. In *Proceedings of Formal Concept Analysis*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 269–284.
9. Burusco, A.; Fuentes Gonzalez, R. The study of the L-fuzzy concept lattice. *Mathware Soft Comput.* **1994**, *1*, 209–218.
10. Burusco, A.; Fuentes Gonzalez, R. The study of the interval-valued contexts. *Fuzzy Sets Syst.* **2001**, *121*, 439–452. [[CrossRef](#)]
11. Dubois, D.; Prade, H. Possibility theory and formal concept analysis: Characterizing independent sub-contexts. *Fuzzy Sets Syst.* **2012**, *196*, 4–16. [[CrossRef](#)]
12. Singh, P.; Aswani Kumar, C. Bipolar fuzzy graph representation of concept lattice. *Inf. Sci.* **2014**, *288*, 437–448. [[CrossRef](#)]
13. Singh, P. Processing linked formal fuzzy context using non-commutative composition. *Inst. Integr. Omics Appl. Biotechnol.* **2016**, *7*, 21–32.
14. Wang, L.; Liu, X. Concept analysis via rough set and AFS algebra. *Inf. Sci.* **2008**, *178*, 4125–4137. [[CrossRef](#)]
15. Yao, Y. A comparative study of formal concept analysis and rough set theory in data analysis. In *Proceedings of the 4th International Conference on Rough Sets and Current Trends in Computing (RSCTC 2004)*, Uppsala, Sweden, 1–5 June 2004; pp. 59–68.
16. Yao, Y.; Mi, J.; Li, Z.; Xie, B. The construction of fuzzy concept lattices based on (θ, δ) -fuzzy rough approximation operators. *Fundam. Informaticae* **2011**, *111*, 33–45. [[CrossRef](#)]
17. Krupka, M.; Laštovička, J. Concept lattices of incomplete data. In *Proceedings of the International Conference on Formal Concept Analysis*, Leuven, Belgium, 7–10 May 2012; pp. 180–194.
18. Li, M.; Wang, G. Approximate concept construction with three-way decisions and attribute reduction in incomplete contexts. *Knowl. Based Syst.* **2016**, *91*, 165–178. [[CrossRef](#)]
19. Li, J.; Mei, C.; Lv, Y. Incomplete decision contexts: Approximate concept construction, rule acquisition and knowledge reduction. *Int. J. Approx. Reason.* **2013**, *54*, 149–165. [[CrossRef](#)]
20. Simiński, K. Neuro-rough-fuzzy approach for regression modelling from missing data. *Int. J. Appl. Math. Comput. Sci.* **2012**, *22*, 461–476. [[CrossRef](#)]
21. Hao, C.; Fan, M.; Li, J. Optimal scale selecting in multi-scale contexts based on granular scale rules. *Pattern Recognit. Artificial Intell.* **2016**, *29*, 272–280. (In Chinese)
22. Ma, L.; Mi, J.; Xie, B. Multi-scaled concept lattices based on neighborhood systems. *Int. J. Mach. Learn. Cybern.* **2017**, *8*, 149–157. [[CrossRef](#)]
23. Wu, W.; Leung, Y. Theory and applications of granular labeled partitions in multi-scale decision tables. *Inf. Sci.* **2011**, *181*, 3878–3897. [[CrossRef](#)]
24. Wu, W.; Leung, Y. Optimal scale selection for multi-scale decision tables. *Int. J. Approx. Reason.* **2013**, *54*, 1107–1129. [[CrossRef](#)]
25. Tang, Y.; Fan, M.; Li, J. An information fusion technology for triadic decision contexts. *Int. J. Mach. Learn. Cybern.* **2016**, *7*, 13–24. [[CrossRef](#)]
26. Godin, R.; Missaoui, R.; Alaoui, H. Incremental concept formation algorithms based on Galois (concept) lattices. *Comput. Intell.* **1995**, *11*, 246–267. [[CrossRef](#)]
27. Zou, L.; Zhang, Z.; Long, J. A fast incremental algorithm for constructing concept lattices. *Expert Syst. Appl.* **2015**, *42*, 4474–4481. [[CrossRef](#)]
28. Zou, L.; Zhang, Z.; Long, J. A fast incremental algorithm for deleting objects from a concept lattice. *Knowl.-Based Syst.* **2015**, *89*, 411–419. [[CrossRef](#)]
29. Shao, M.; Leung, Y. Relations between granular reduct and dominance reduct in formal contexts. *Knowl.-Based Syst.* **2014**, *65*, 1–11. [[CrossRef](#)]
30. Wei, L.; Qi, J.; Zhang, W. Attribute reduction theory of concept lattice based on decision formal contexts. *Sci. China Ser. F-Inf. Sci.* **2008**, *51*, 910–923. [[CrossRef](#)]
31. Li, J.; Mei, C.; Lv, Y. Knowledge reduction in decision formal contexts. *Knowl.-Based Syst.* **2011**, *24*, 709–715. [[CrossRef](#)]
32. Li, J.; Mei, C.; Lv, Y. Knowledge reduction in real decision formal contexts. *Inf. Sci.* **2012**, *189*, 191–207. [[CrossRef](#)]
33. Li, J.; Mei, C.; Aswani Kumar, C.; Zhang, X. On rule acquisition in decision formal contexts. *Int. J. Mach. Learn. Cybern.* **2013**, *4*, 721–731. [[CrossRef](#)]

34. Shao, M.; Leung, Y.; Wu, W. Rule acquisition and complexity reduction in formal decision contexts. *Int. J. Approx. Reason.* **2014**, *55*, 259–274. [[CrossRef](#)]
35. Wei, L.; Li, T. Rules acquisition in consistent formal decision contexts. In Proceedings of the 11th International Conference on Machine Learning and Cybernetics (ICMLC'12) Xi'an, China, 15–17 July 2012; pp. 801–805.
36. Wu, W.; Leung, Y.; Mi, J. Granular computing and knowledge reduction in formal contexts. *IEEE Trans. Knowl. Data Eng.* **2009**, *21*, 1461–1474.
37. Qi, J.; Wei, L.; Yao, Y. Three-way formal concept analysis. In Proceedings of the 2014 International Conference on Rough Sets and Knowledge Technology, Shanghai, China, 24–26 October 2014; pp. 732–741.
38. Qi, J.; Qian, T.; Wei, L. The connections between three-way and classical concept lattices. *Knowl.-Based Syst.* **2016**, *91*, 143–151. [[CrossRef](#)]
39. Qian, T.; Wei, L.; Qi, J. Constructing three-way concept lattices based on apposition and subposition of formal contexts. *Knowl.-Based Syst.* **2017**, *116*, 39–48. [[CrossRef](#)]
40. Wang, W.; Qi, J. Algorithm for constructing three-way concepts. *J. Xidian Univ.* **2017**, *44*, 71–76.
41. Yang, S.; Lu, Y.; Jia, X.; Li, W. Constructing three-way concept lattice based on the composite of classical lattices. *Int. J. Approx. Reason.* **2020**, *121*, 174–186. [[CrossRef](#)]
42. Yao, Y. Three-way decisions with probabilistic rough sets. *Inf. Sci.* **2010**, *180*, 341–353. [[CrossRef](#)]
43. Yao, Y. Three-way decision and granular computing. *Int. J. Approx. Reason.* **2018**, *103*, 107–123. [[CrossRef](#)]
44. Yu, H.; Li, Q.; Cai, M. Characteristics of three-way concept lattices and three-way rough concept lattices. *Knowl.-Based Syst.* **2018**, *146*, 181–189. [[CrossRef](#)]
45. Li, J.; Xu, W.; Qian, Y. Concept learning via granular computing: A cognitive viewpoint. *Inf. Sci.* **2015**, *298*, 447–467. [[CrossRef](#)]
46. Li, J.; Huang, C.; Qi, J.; Qian, Y.; Liu, W. Three-way cognitive concept learning via multi-granularity. *Inf. Sci.* **2017**, *378*, 244–263. [[CrossRef](#)]
47. Bělohlávek, R.; De Baets, B.; Konecny, J. Granularity of attributes in formal concept analysis. *Inf. Sci.* **2014**, *260*, 149–170. [[CrossRef](#)]
48. Hashemi, R.; Agostino, S.; Westgeest, B.; Talburt, J. Data granulation and formal concept analysis. In Proceedings of the Processing NAFIPS—04. IEEE Annual Meeting of the Fuzzy Information, Banff, AB, Canada, 27–30 June 2004; Volume 1, pp. 79–83.
49. Zou, L.; Zhang, Z.; Long, J. An efficient algorithm for increasing the granularity levels of attributes in formal concept analysis. *Expert Syst. Appl.* **2016**, *46*, 224–235. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.