Studies on Several Methods for Integrating Formal Concept Analysis and Statistical Machine Learning

Siqi Peng

Department of Intelligence Science and Technology
Graduate School of Informatics, Kyoto University

This dissertation is submitted for the degree of
Doctor of Philosophy (Informatics)

September 2024

List of Publications

Publications related to this thesis

The content of this thesis contains the following publications. All of these are journal papers.

Chapter 2 contains the following journal paper. The paper is granted the *IPSJ Specially Selected Paper of FY2023*.

[J1] Peng, S., & Yamamoto, A. (2023). Z-tca: fast algorithm for triadic concept analysis using zero-suppressed decision diagrams. *Journal of Information Processing*, 31:722-733.

Chapter 3 contains the following journal paper. The paper is accepted by *Journal of Information Processing* at the time point when the thesis is completed and pending for publication, so the page numbers are not listed.

[J2] Peng, S., & Yamamoto, A. (2024). Concept lattice reduction using integer programming. *Journal of Information Processing*, 32.

Chapter 4 contains the following journal paper.

[J3] Peng, S., Yamamoto, A., & Ito, K. (2023). Link prediction on bipartite networks using matrix factorization with negative sample selection. *PLOS ONE*, 18(8):e0289568.

Chapter 5 contains the following journal paper. The paper is a co-work by Siqi Peng, the author of this thesis and another researcher Hongyuan Yang, a master's student of Graduate School of Informatics, Kyoto University at the time point when this thesis is completed. Both contributors are supervised by the supervisor Akihiro Yamamoto.

[J4] Peng, S., Yang, H., & Yamamoto, A. (2024). BERT4FCA: A method for bipartite link prediction using formal concept analysis and BERT. *PLOS ONE*, 19(6): e0304858.

The contributions of two contributors to the above paper are listed below:

- HY proposes the framework to use a modified BERT network to capture the information within extents/intents and the order relations between extents/intents, which corresponds to the contents related on O-O task in the publication.
- SP proposes the framework to use two modified BERT networks proposed by HY to capture the information of extents and intents, separately, and combine the pre-trained models in the network for fine-tuning, which corresponds to the contents on the O-A task in the publication.

Both contributors have agreed that they contribute equally to this work. In the original publication of this work, both contributors are in the author list and are marked as equally contribution.

Other Publications

Besides the above, the author of this thesis has the following publications and during his PhD program. Note that the non-peer-reviewed publications that have been extended and updated into peer-reviewed journal papers are omitted in the list.

Peer-reviewed Conference Papers (Workshops included)

[C1] Peng, S., Yamamoto, A., Mori, S., & Sekino, T. (2022, December). Event time extraction from Japanese news archives. In *2022 IEEE International Conference on Big Data (Big Data)* (pp. 2556-2564). IEEE.

Acknowledgements

First of all, I would like to acknowledge my deepest gratitude to my supervisor, Prof. Akihiro Yamamoto. His instructions, advice, and support not only helped me acquire the knowledge and skills required as a researcher, but also granted me a deeper understanding of questions like “what is computer science” and “what are the purposes of scientific research”. These will have a deep influence on my future career as a researcher.

Secondly, I would like to express my gratitude to all of my co-researchers. They include Hongyuan Yang, who participated in the co-research on BERT4FCA; Prof. Kimihito Ito, who offered instructions and suggestions to my research on MF-NSS; Prof. Ken Satoh and Prof. Yoshiaki Okubo, who gave comments and support to my research on the ILP-based concept lattice reduction method as well as MF-NSS; Prof. Shinsuke Mori and Prof. Tatsuki Sekino, who shared their experiences in natural language processing with me. I would never have completed these studies without the help of my co-researchers.

Thirdly, I would like to express my gratitude towards Prof. Hisashi Kashima and Prof. Tatsuya Akutsu, who participated in the examination for my doctoral degree. Their suggestions and comments have helped increase the readability and the scientific rigor of this thesis.

Moreover, I would like to express my gratitude to all organizations that grant me financial support during my PhD program. These include the Japan Society for the Promotion of Science (JSPS), the Creation and Organization of the Algorithmic Foundations for Social Advancement (AFSA) project led by Prof. Shin-ichi Minato, and the Kyoto University Division of Graduate Studies. The scholarships they offered had made me able to devote myself in researching activities, and the research expenses they sponsored had greatly improved the quality of my research experience.

Last but not least, I would like to express my gratitude to my parents, who continuously gave me encouragement and financial support throughout the years of my PhD program.

Abstract

Formal concept analysis (FCA) is a technique for knowledge extraction from *formal contexts* consisting of binary relations of a group of objects and their attributes. The extracted knowledge is in the form of *formal concepts*, which represent strongly related groups objects and their attributes. All extracted knowledge can be organized into a *concept lattice* by their conceptual hierarchical relations. FCA has found practical applications in various fields in data mining and knowledge processing.

Besides the basic applications, FCA can also be applied to *machine learning* tasks by integrating it with a machine learning method. A machine learning task is the task that predicts the properties of the unseen part of some data based on the observed part of the data. Various studies have shown that the integration of FCA can contribute to better performance, thanks to the latent information within the formal concepts and the concept lattice.

The method integrating FCA and a machine learning method can be roughly divided into two groups. One being *FCA4RML*, which integrates FCA with a *rule-based machine learning* (RML) method, and the other one is *FCA4SML*, which integrates FCA with a *statistical machine learning* (SML) method. Studies have proven that SML methods have better capability to capture the information within the data and make generalizations. It has also been proven that concept lattices contain important information which is helpful for various machine learning tasks. Hence, we consider it is of high values to integrate FCA with an SML method and expect the integrated method to have a very high performance. However, in the recent five years, only two FCA4SML methods have been proposed. Although both methods have initially shown good performance, we consider that these methods are still in the early stage of development because these are outperformed by state-of-the-art methods by a large margin.

In order to follow up their research and seek for new breakthroughs in FCA4SML, in this thesis, we present three issues causing it difficult to develop such an FCA4SML method, and propose four solutions dealing with these issues. The issues and our proposed solutions are listed as follows.

First, the FCA process itself requires much running time. In such a long time, some fast SML methods can already give the final output of a high performance, while an FCA4SML

method still need to post-process these extracted formal concepts. To solve this issue, we propose a fast algorithm for FCA and its natural three-dimensional extension, *triadic concept analysis* (TCA). Experimental results have shown that the method is three times faster than previous methods in dense contexts.

Second, the size of the concept lattice is usually too large to be processed by an SML method. To solve this issue, we propose two solutions applying different strategies. We first propose a generally-applicable reliable method to reduce the concept lattice without excessively changing the content of the context and the concept lattice. We then propose a task-specified FCA4SML framework which can utilize the information provided by FCA without directly processing the full concept lattice with the SML method.

Third, there are two types of information in the concept lattice – the relations of objects/attributes within formal concepts and the conceptual hierarchy of concepts, making it hard to develop a method which can efficiently capture both types of information in order to make full use of FCA. To solve this issue, we develop an FCA4SML framework called *BERT4FCA*, which uses a *BERT*-like *Transformer encoder* network to fully capture both types of information. Experimental results have shown that our method can indeed capture the information and capturing the information indeed contributes to a high performance.

Among the four methods we propose in the thesis, two of these are general improvement to FCA, and the other two methods are counted as the most high-performance methods in the machine learning tasks of *link prediction* and *hyper-link prediction* on bipartite networks. Hence, we believe the methods we propose will not only contribute to research on FCA4SML, but also open up new possibilities to the research on FCA itself and all related machine learning tasks studied in this thesis.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 General information on our contribution

Formal concept analysis (FCA) is a method for extracting and analyzing closed relations called *formal concepts* from a *formal context* consisting of a collection of *objects* and their *attributes*. FCA was first proposed by Ganter and Wille in the 1980s [112] in order to find a real-world application for the abstract *lattice* structure studied in mathematical *order theory* [21, 113]. Over the years, FCA has then found its applications in various practical fields including *text mining* [17, 59], *software engineering* [11, 14, 30, 80], and *medical and biological ontology engineering* [31, 67, 82, 111]. Currently, FCA has become one of the most important topics in the field of data mining and knowledge processing [92, 93, 95].

The input of FCA is a relational data table called a *formal context*, which represents the binary relations of two groups of variables called *objects* and *attributes*. In Fig. 1.1, we show a sample formal context in the form of a cross table [1]. In the table, each row represents an *object* and each column represents an *attribute*. In this specific example, the objects are the diseases, and the attributes are the symptoms of the diseases. Each cross in the table indicates that a binary relation holds between the object and the attribute. In this specific example, it means a person affected by the corresponding disease will show the corresponding symptom. Given such a formal context, the goal of FCA is to extract *formal concepts*, defined as maximal subsets of objects and the corresponding maximal attributes they share in common. Based on the definition, in the cross table, the formal concepts may be represented by maximal rectangular boxes filled with crosses with the rows and columns of the table permutable. For example, the formal concept represented by the shaded rectangular box in the aforementioned table means that the maximal group of

[1] The content of this table does not assure real-world accuracy.

diseases having the symptoms of dry cough, fatigue, and fever includes flu and Covid-19, and the maximal group of symptoms that both flu and Covid-19 have includes dry cough, fatigue, and fever. Such a formal concept is considered to represent a closed related group of objects and attributes, where "closed" means the relation breaks when either a new object or a new attribute is inserted into the group. For example, the aforementioned formal concept represents the closed relation of two diseases, flu, Covid-19 and three symptoms, dry cough, fatigue, and fever. If a new disease is inserted into the group, the three symptoms in the group will not be possessed by all diseases in the group, and thus the relation breaks; similarly, if a new symptom is added to the group, the two diseases will not have all symptoms in the group and thus the relation no longer holds. Such closed relation is considered to be an important form of knowledge, especially in the field of ontology and taxonomy engineering [113].

	Body Aches	**Chills**	**Dry Cough**	**Fatigue**	**Fever**	**Headache**
Allergies			×	×		×
Cold	×		×	×		
Flu		×	×	×	×	×
Covid-19			×	×	×	

Fig. 1.1 A sample formal context of symptoms of some diseases.

In the left panel of Fig. 1.2, we show another disease-symptom formal context. In this context, for convenience, the names of diseases are replaced with symbols $g_1, g_2, \cdots, g_8$, and the names of symptoms are replaced with symbols $m_1, m_2, \cdots, m_5$. With such symbolized names, we can conveniently represent a formal concept with a pair of sets. For example, the shaded rectangular box in the cross table of the aforementioned sample context can be represented by $(\{g_2, g_3\}, \{m_1, m_2, m_3\})$. From the sample context, we can extract in total 13 concepts. If we list them out, we may find some of these concepts have a conceptual hierarchical relation. For example, in the context, we can extract another concept $(\{g_1, g_2, g_3\}, \{m_2, m_3\})$. Compared to the aforementioned concept $(\{g_2, g_3\}, \{m_1, m_2, m_3\})$, this concept has one more attribute m_1, but lacks one object g_1. In the specific example where objects represent diseases and attributes represent symptoms, the relation between two concepts will mean that "if we add Symptom m_1 into the group, the maximal group of diseases having all symptoms in the group will reduce from Diseases g_1, g_2 and g_3 to Diseases g_2 and g_3". Such hierarchical relations of concepts are also considered important because they represent the "levels" of the knowledge [21, 113]. To show these hierarchical relations better, we may organize the concepts into a *concept lattice*, which can be figuralized into a line diagram as is shown in the right panel of Fig. 1.2. Note that this diagram is shown in a minimally

annotated form. A fully annotated version of the diagram where every node is annotated with its corresponding concept is shown in Fig. 1.3. The instructions on how to read the minimally annotated version of the diagram will be later introduced in Section 1.2. In this thesis, we prefer to use the minimally annotated form in order to keep it concise.

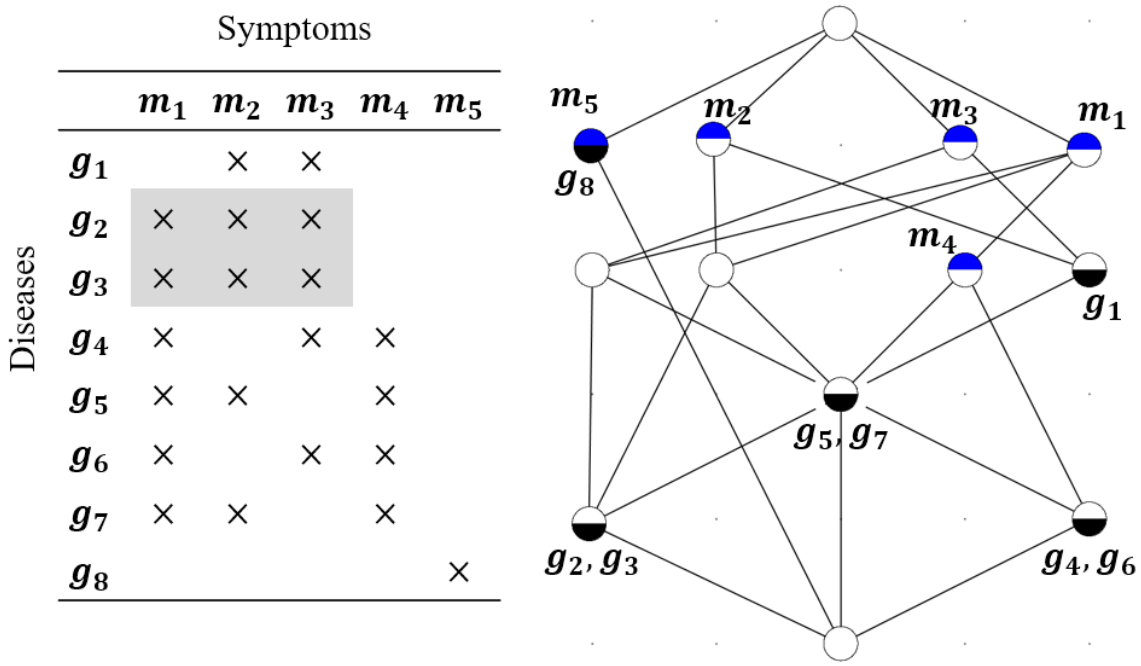


Fig. 1.2 Left: A sample formal context of symptoms of some diseases in an abstract form. Right: The concept lattice corresponding to the context in the left panel. The concept lattice is shown in the form of a line diagram. Please refer to Section 1.2 for how to read the diagram. The fully annotated form of the diagram is shown in Fig. 1.3.

Sometimes in our real-world life, we may deal with relations among more than two groups of variables. In this case, we can generalize FCA to a high-dimensional version. For example, if we add another group of variables *conditions* in addition to the objects and attributes, we may extend the formal context into its three-dimensional variation called a *triadic formal context* or shortly *triadic context*. In Fig. 1.4, we show a sample triadic context demonstrating the symptoms of a group of diseases with different levels of severity [2]. In this example, a cross should represent that a person affected with a disease with the corresponding level of severity has the corresponding symptom. For example, Given such a triadic concept, we can extract the three-dimensional version of formal concepts, called *triadic concepts*, which are defined as those maximal cubic rectangular boxes filled with

[2]The content of this table does not assure real-world accuracy.

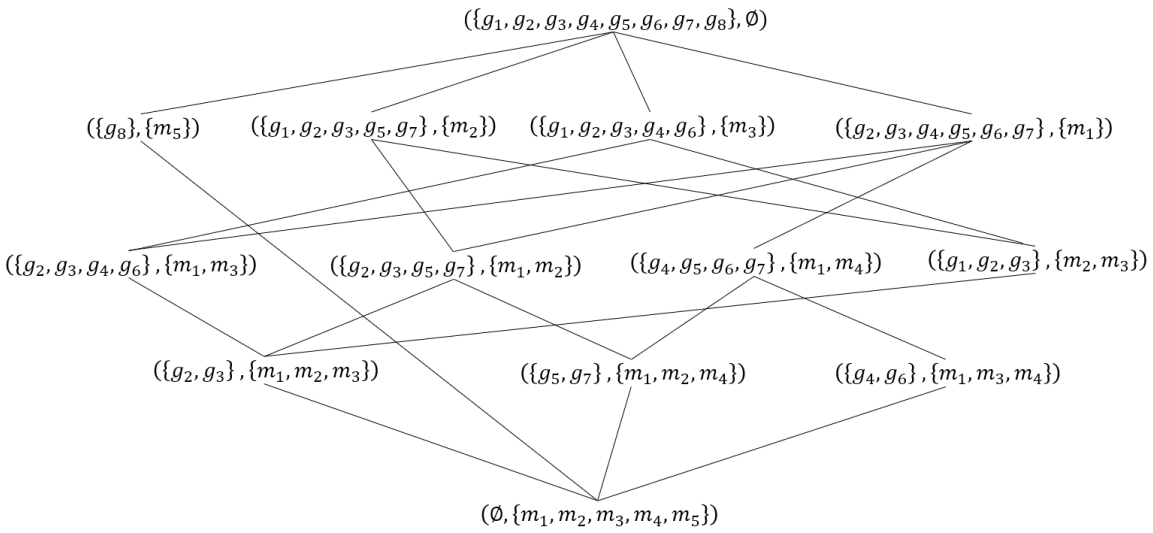


Fig. 1.3 The fully annotated diagram of the concept lattice corresponding to the context in the left panel of Fig. 1.2.

crosses with the rows, columns, and aisles[3] permutable. For example, the triadic concept represented by the shaded cubic box in the table means that the maximal group of diseases that have the symptoms of fatigue at both the mild level and the severe level includes cold, flu, and Covid-19. The task of extracting triadic concepts from triadic contexts is named *triadic concept analysis* (TCA), which is considered the natural generalization of FCA and has become an important sub-topic in the general field of FCA [64]. Note that, unlike the case in the basic two-dimensional FCA, the collection of all triadic concepts from a formal context cannot be organized into a mathematical lattice structure, so the research on TCA mainly focuses on extracting the triadic concepts, instead of extracting and organizing them. Certainly, we may further generalize TCA into a higher-dimensional version called *polyadic concept analysis* [8, 108].

In the early times, FCA was mainly usually used for *knowledge representation* and *data visualization* because the formal concepts were considered to be a form of representing the real-world knowledge [21, 37, 112, 113]. Starting from the 1990s, researchers noticed that the formal concepts and their hierarchical orders actually contain more latent information, so they started to apply FCA to *data mining* and *knowledge discovery* tasks [11, 46, 92, 93, 95], aiming at extracting useful information from the data, *i.e.*, formal contexts. Researchers then found that the extracted information may be further processed to fulfill more complex downstream tasks, so they started to apply FCA to more complex *machine learning* tasks, aiming at inferring the **unseen** part of the data or predicting the **unobserved** knowledge

[3]The aisles refers to the one-dimensional section of a cubic matrix orthogonal to both rows and columns.

	Mild				**Severe**			
	Body Aches	**Chills**	**Dry Cough**	**Fatigue**	**Body Aches**	**Chills**	**Dry Cough**	**Fatigue**
Allergies			×				×	×
Cold	×		×	×	×	×		×
Flu		×	×	×		×	×	×
Covid-19			×	×	×	×	×	×

Fig. 1.4 A sample triadic context of symptoms of some diseases with different levels of severity.

from the **observed** part of the data [9, 34, 36, 93, 121]. Especially note that the FCA process itself can only extract formal concepts and construct concept lattices. To generate predictive functions, the constructed concept lattices still need to be further processed by other methods. The method used for post-processing the concept lattices is usually a *rule-based machine learning* (RML) method, which gives predictions to the unseen part or features of the data with rules determined by human experts. For example, in [121], the authors applied the following rule for predicting new object-attribute relations in the formal context. For two concepts overlapping with each other, if the ratio of the overlapped part exceeds a predefined threshold, they are considered parts of the same large formal concept, so all the missing relations in the non-overlapped part *a.k.a. the structure hole* [63] are predicted as new relations. We use the term *FCA4RML methods* for referring to the methods for solving machine learning tasks by integrating FCA with an RML method.

With the rapid development of machine learning theory, newly developed *statistical machine learning* (SML) methods have soon outperformed most RML methods [68, 119], including FCA4RML methods [28, 69]. Unlike RML methods, an SML method does not use human experts' pre-determined rules to give predictions. Instead, it assumes that the data follow some form of predictive function, and then uses *statistical inference* methods to estimate the parameters of the function [68]. Thanks to the rapid increase of computing power, the number of parameters used in the predictive functions of SML methods has grown sharply in the past ten years, making these methods have far better capabilities in fitting the observed part of the data and generalizing to the unseen part [68, 119]. Hence, generally, SML methods have far better performance than RML methods [68, 119]. Based on the fact that FCA4RML methods have integrated FCA to boost the performance of the RML method, we have naturally come up with the idea to integrate FCA with better-performing SML methods. We name these integrated methods *FCA4SML* method and expect them to have better performance than the SML methods without integration of FCA because

the concept lattice is a more well-structuralized form of data compared to the raw formal context, and previous research has shown that SML methods can capture features from such well-structuralized data more easily than raw data [13, 73, 101, 110].

Despite that FCA4SML methods are expected to have a good performance, till now, very few FCA4SML methods have been worked out. To the furthest of our knowledge, except the work of the author of this thesis, **only two** FCA4SML methods have been proposed in the past five years.

- In [28], the authors proposed *object2vec* which aims to embed objects into vectors using the information from the formal concepts. It has two embedding models, *object2vec-CBoW* and *object2vec-SG*, both are derived from *Word2Vec* [75, 76]. Object2vec-CBoW, based on the *continuous-bag-of-words* model from Word2Vec, predicts a target object using objects around it within the same concept; object2vec-SG, based on the *skip-gram* model from Word2Vec, uses an object to predict other objects in the same concept. They conducted experiments on the task of predicting new co-authorship in an author-publication network and demonstrated good performance.

- In [69], the authors proposed another embedding method called *Bag of Attributes (BoA)*. Their method trains a more complicated embedding model using *Bidirectional Long Short-Term Memory* (Bi-LSTM) [39] and *Variational Autoencoder* (VAE) [53] on formal contexts. They conducted experiments on the same task as the previous research, and the results are similar to those obtained with object2vec.

We found these two methods interesting because they have shown the possibility of working out an FCA4SML method. Moreover, their performances are already far better than FCA4RML methods and approach some classic non-FCA-based SML methods like *node2vec*, which have initially proven the great value of integrating FCA with SML. However, these methods are still far from practically applicable because their performances in the tasks studied in these two works are still outperformed by state-of-the-art SML methods by a large margin [45, 72]. We have summarized three points that make it difficult to develop a practically applicable FCA4SML method.

- The time complexity of FCA itself, *i.e.*, the process that extracts all formal concepts, is high. Extracting all concepts from a medium-sized context may cost several hours. In such a long time, some fast SML methods can already give the final output of a high performance, while an FCA4SML method still needs post-processing to these extracted formal concepts.

- The volume of formal concepts to be post-processed is extremely large. SML methods usually work in an iterative optimization mode, which needs to process the data for

multiple epochs. Hence, a small increase to the input data will result in a large increase in running time.

- The information within the concept lattice is complicated. There are two types of information within a concept lattice – the relations between objects/attributes within a formal concept, and the order relations between formal concepts. It is difficult to design a mechanism that can effectively capture both types of information to make full use of FCA.

Aiming at solving these issues and bringing new possibilities to the studies of FCA4SML methods, in this thesis, we will propose the following four solutions, each of which will be introduced in a separate chapter.

- To solve the first issue, we propose *Z-TCA*, a fast algorithm for FCA and TCA, *a.k.a.*, the three-dimensional natural extension of FCA. The algorithm utilizes a special data structure *zero-suppressed decision diagrams* (ZDD) to speed up the set operations. We conducted experiments and the results have shown that our algorithm is about three times as fast as previous methods in denser contexts.

- For the second issue, we propose the following two solutions.

  – We propose a method for reducing the number of objects as well as formal concepts by adjusting the data using *integer linear programming*. Compared to previous methods, our method is more reliable thanks to its ability to prevent the context and the concept lattice from being modified too much. Such reliability is especially important for FCA4SML methods because the SML process relies strongly on the input data to fit the weights in the predictive model, so unreliable input data will have a significant negative influence on the performance.

  – We propose an FCA4SML method that avoids direct post-processing extracted formal concepts using SML methods. Instead, our method first uses a rule-based process called *negative sample selection* to post-process the extracted formal concepts to initialize the training samples for the SML process. Then, our method applies the SML process to train a model using the samples generated with the help of FCA. We apply this method to a practical bipartite link prediction task and the experimental results have shown that the special process of negative sample selection contributes to better performance, showing the possibility to utilize the information provided by FCA without direct post-processing extracted formal concepts using SML methods.

- For the third issue, we propose an FCA4SML method integrating FCA and *BERT* [24, 29], a framework that utilizes a stacked *Transformer encoder* [107] network to train large language models. We found that the input BERT takes is very similar to the formal concepts, so we modified the network structure of BERT, making it capable of efficiently capturing both types of information from concept lattices. We apply this method to two practical tasks on bipartite networks, and the experimental results have shown that our method has a high performance. We have also conducted ablation experiments to show that our method has indeed captured both types of information in the concept lattice, and the information has indeed contributed to the good performance of the method. Note that this is a co-work by Siqi Peng, the author of this thesis, and another researcher Hongyuan Yang. The contributions of two co-authors will be presented in Section 5.1.

The rest part of this thesis is organized as follows. In the rest part of Chapter 1, we are going to introduce the preliminary knowledge of FCA and its natural extension TCA. In Chapters 2 to 5, we are going to introduce each of our contributions in a separate chapter. In Chapter 6, we will draw the conclusions of our contributions and discuss our future work. Finally in the appendix, we are to give a running example of the algorithm we are to introduce in Chapter 3, so that the readers can understand it better.

1.2 Mathematical basics of formal concept analysis (FCA)

In this section, we introduce the mathematical preliminaries of the basic form of FCA.

FCA takes *formal contexts* as input. A formal context is defined as follows.

Definition 1. *A **formal context** is a triple $\mathbb{K} := (G, M, Y)$, where G is a set of objects, M is a set of attributes, and $Y \subseteq G \times M$ is a binary relation called incidence that expresses which object has which attribute. We write gYm or $(g,m) \in Y$ to express that the object $g \in G$ has the attribute $m \in M$.*

Formal contexts are illustrated in cross tables, as exemplified in the left of Fig. 1.2, where rows correspond to objects and columns to attributes, and a cell is marked with a cross if the object in its row has the attribute in its column. In the context shown in the left panel of Fig. 1.2, the marked cell represents that the object listed in the row possesses the corresponding attribute in the column.

From the formal context, FCA aims to extract *formal concepts*. They are defined as below.

Definition 2. *In a formal context $\mathbb{K} = (G, M, Y)$, for subsets of objects and attributes $A \subseteq G$ and $B \subseteq M$, (A, B) is called a **formal concept** if $\forall (A_1, B_1)$ such that $A \subseteq A_1 \subseteq G$ and $B \subseteq B_1 \subseteq M$, $A_1 \times B_1 \subseteq Y$ is satisfied if and only if $A = A_1$ and $B = B_1$.*

Definition 3. *In a formal concept (A, B), A is defined as the **extent** of the concept, and B is defined as the **intent** of the concept. A and B may also be directly called an **extent** and an **intent**, correspondingly.*

In the cross-table form of a formal context, a formal concept can be represented with a maximal rectangular box filled up with crosses. For example, the formal concept corresponding to the shaded rectangle in the context shown in the left panel of Fig. 1.2 is $(\{g_2, g_3\}, \{m_1, m_2, m_3\})$.

An important feature of formal concepts is that one can generate a formal concept from a subset of attributes or objects, with *derivation operators*.

Definition 4. *Given a formal context $\mathbb{K} = (G, M, Y)$, an object set A_1 and an attribute set A_2, the **derivation operators** $A_1^{(2)}$ and $A_2^{(1)}$ is defined as below:*

$$A_1^{(2)} = \{a_2 \mid \forall a_1 \in A_1,\ (a_1, a_2) \in Y\},$$
$$A_2^{(1)} = \{a_1 \mid \forall a_2 \in A_2,\ (a_1, a_2) \in Y\}.$$

Given an attribute subset $X \subseteq M$, one can generate a formal concept $(X^{(1)}, X^{(1)(2)})$, where $X^{(1)(2)}$ means the combination of two derivation operators, *i.e.*, $(X^{(1)})^{(2)}$. For example, in the formal context shown in the left panel of Fig. 1.2, suppose we start from the attribute set $X = \{m_2, m_3\}$, we may first generate $X^{(1)}$, that is, all diseases that have Symptoms m_2 and m_3, which would be $\{g_1, g_2, g_3\}$. Then, we are to compute $X^{(1)(2)}$, that is, the symptoms that Diseases $\{g_1, g_2, g_3\}$ all have, and this would be $\{m_2, m_3\}$. Now we get the formal concept $(\{g_1, g_2, g_3\}, \{m_2, m_3\})$. Most algorithms for FCA use this derivation operator for generating new concepts or checking whether a tuple is a concept [37, 112].

After all concepts are extracted, we can organize them into a mathematical *lattice* structure called *concept lattice*. It is defined as follows.

Definition 5. *Given a context $\mathbb{K} = (G, M, Y)$, the **concept lattice** of context $\mathbb{K}$, denoted by $\underline{\mathfrak{B}}(\mathbb{K})$, is the structure that organizes the set of all concepts extracted from context $\mathbb{K}$ with the hierarchical order $\leq$. For two concepts (A_1, B_1) and (A_2, B_2), we write $(A_1, B_1) \leq (A_2, B_2)$ if $A_1 \subseteq A_2$ (which mutually implies $B_2 \subseteq B_1$). In such a case, (A_1, B_1) is called an **infimum** of (A_2, B_2) and (A_2, B_2) is called a **supremum**[4] of (A_1, B_1).*

[4]The plural form of infimum is *infima*. The plural form of supremum is *suprema*.

Concept lattices are usually figuralized with line diagrams. For example, the line diagram shown in the right panel of Fig. 1.2 represents the concept lattice of the context represented in the left panel of the same figure. In the diagram, nodes represent formal concepts and lines represent hierarchical orders. Each node in the diagram can be marked with at most two labels – the label marked above the node is called the upper label, and that marked below the node is called the lower label. The upper label of a node should always be a set of attributes, and the lower label of a node should always be a set of objects. If a node does not have an upper/lower label, its upper/lower label is considered to be $\emptyset$. For each node, say node X in the diagram, suppose that the upper labels of the nodes on the path from the top-most node to node X (including the top-most node and node X) are $M_1, M_2, M_3, \cdots, M_x$, and the lower labels of the nodes on the path from the bottom-most node to node X (including the bottom-most node and node X) are $G_1, G_2, G_3, \cdots, G_y$. Then, the formal concept corresponding to node X is $(G_1 \cup G_2 \cup \ldots \cup G_y, M_1 \cup M_2 \cup \ldots \cup M_x)$.

A special case of the order relation is the *neighboring relation*, which is defined as follows.

Definition 6. *In a concept lattice $\underline{\mathfrak{B}}(\mathbb{K})$, for two concepts (A_1, B_1) and (A_2, B_2) such that $(A_1, B_1) < (A_2, B_2)$, (A_1, B_1) is defined to be the **lower neighbor** of (A_2, B_2) if $\nexists (A_3, B_3) \in \underline{\mathfrak{B}}(\mathbb{K})$ such that $(A_1, B_1) < (A_3, B_3) < (A_2, B_2)$. In this case, (A_2, B_2) is dually called the **upper neighbor** of (A_1, B_1) or we may also directly say that (A_1, B_1) and (A_2, B_2) are **neighbors** or that they have the **neighboring relation**.*

In the line diagram representation of a concept lattice, if two concepts are neighbors, they will be shown as literally "neighbors", that is, the node corresponding to the concepts will be directly connected by a line.

1.3 Mathematical basics of triadic concept analysis (TCA)

As introduced in the previous subsection, FCA deals with binary relations of two groups of elements – the objects and the attributes. In the real world, however, sometimes we may need to deal with ternary relations of three groups of elements. In such case, we can naturally generalize the theory of FCA into a three-dimensional version, *triadic concept analysis* (TCA).

When we discuss about TCA, the terms related to the basic form of FCA may be emphasized with "dyadic". For example, the *dyadic* formal context means the basic binary relational formal context.

TCA deals with *triadic formal contexts*, the three-dimensional version of *formal contexts*. They are defined as follows.

Definition 7. *A **triadic formal context**, shortly **triadic context** is a quadruple (G,M,B,Y) where G,M,B are sets and Y is a set of ternary relations between G,M,B, i.e., $Y \subseteq G \times M \times B$. Mathematically G,M, and B are homogeneous, each of them often being referred to as a **component** or a **dimension** of the context; in practice, however, G,M, and B are often considered as the set of **objects**, **attributes**, and **conditions**, and a ternary relation $(g,m,b) \in Y$ means object g has attribute m under the condition b.*

Triadic context should be represented by "cubic" cross tables. Since it is difficult to draw a real cubic table, practically, the depth of the table is usually flattened. A sample cross table representation of a triadic context with the names of objects, attributes, and conditions represented by symbols and the depths of the table flattened is shown in 1.5.

Stages		b_1				b_2			
Symptoms		m_1	m_2	m_3	m_4	m_1	m_2	m_3	m_4
Diseases	g_1						×		
	g_2	×	×		×	×			×
	g_3	×	×		×	×			×
	g_4	×			×		×		×
	g_5	×		×				×	
	g_6	×		×				×	
	g_7	×		×				×	

Fig. 1.5 A sample triadic context of symptoms of some diseases at different stages.

From a triadic context, TCA aims at extracting *triadic formal concepts*, which are defined as follows:

Definition 8. *A **triadic formal concept**, shortly **triadic concept** of a triadic context (G,M,B,Y) is a triple (A_1,A_2,A_3) where $A_1 \subseteq G, A_2 \subseteq M, A_3 \subseteq B$ and the triple is maximal with respect to dimension-wise set inclusions. That is, $A_1 \times A_2 \times A_3 \subseteq Y$ and for all $X_1 \supseteq A_1, X_2 \supseteq A_2, X_3 \supseteq A_3$ such that $X_1 \times X_2 \times X_3 \subseteq Y$, we should always have $X_1 = A_1, X_2 = A_2$ and $X_3 = A_3$ satisfied.*

Definition 9. *In a triadic concept (A_1,A_2,A_3), the three sets A_1, A_2, and A_3 are called the **components** of the concept. Specifically, A_1 is called the **extent**, A_2 is called the **intent**, and*

*A_3 is called the **modus**[5] of the concept. A_1, A_2 and A_3 may be also directly called an extent, an intent, and a modus, respectively.*

If we represent the triadic context with a three-dimensional cross table, a triadic concept can be represented with a maximal cube-box filled up with crosses, just as shown in Fig. 1.5.

We can use *derivation operators* to generate a triadic concept from a subset of a dimension.

Definition 10. *Given a triadic context $\mathbb{K} = (K_1, K_2, K_3, Y)$. For $X \subseteq K_1$ and $Z \subseteq K_2 \times K_3$ and $i, j, k \in \{1,2,3\}$, $i \neq j \neq k$ and $j < k$, the i-derivation of X and Z, denoted as $X^{(i)}$, $Z^{(i)}$ respectively, are defined to be:*

$$X^{(i)} = \{(a_j, a_k) \in K_j \times K_k \mid \forall a_i \in X,\ (a_i, a_j, a_k) \in Y\},$$
$$Z^{(i)} = \{a_i \in K_i \mid \forall (a_j, a_k) \in Z,\ (a_i, a_j, a_k) \in Y\}.$$

These operators also yield the derivation operators of the context. For a triadic context $\mathbb{K} = (K_1, K_2, K_3, Y)$ and $i, j, k \in \{1,2,3\}$ satisfying $i \neq j \neq k$ and $j < k$, the i-derivation of context $\mathbb{K}$, denoted as $\mathbb{K}^{(i)}$, is defined as:

$$\mathbb{K}^{(i)} = (K_i, K_j \times K_k, \{(a_i, (a_j, a_k)) \mid (a_i, a_j, a_k) \in Y\}).$$

Especially note that neither $X^{(i)}$ nor Z need to be a cartesian product of a $B_j \subseteq K_j$ and $B_k \subseteq X_k$. For example, in the context shown in Table 1.4, let $X = \{g_4\}$ and we should have $X^{(1)} = \{(\{b_1, m_1\}, \{b_1, m_4\}), (\{b_2, m_2\}), (\{b_2, m_4\})\}$, *i.e.*, a set of all (a_2, a_3) pairs satisfying that Disease g_4 has Symptom a_2 in Stage a_3. Clearly, this can not be represented by a cartesian product of an attribute set B_2 and a condition set B_3 because the symptoms of the disease in different stages are different. Hence, unlike the basic FCA case, these two derivation operators are NOT enough to generate a triadic concept.

To generate a triadic concept, we still need the following two derivation operators, denoted as $X_i^{(i,j,A_k)}$ and $X_j^{(i,j,A_k)}$, where $X_i \in K_i, X_j \in K_j$ and $A_k \in K_k$ for $i, j, k \in \{1,2,3\}, i \neq j \neq k$ and $j < k$:

$$X_i^{(i,j,A_k)} = \{a_j \in K_j \mid \forall (a_i, a_k) \in X_i \times A_k,\ (a_i, a_j, a_k) \in Y\},$$
$$X_j^{(i,j,A_k)} = \{a_i \in K_i \mid \forall (a_j, a_k) \in X_j \times A_k,\ (a_i, a_j, a_k) \in Y\}.$$

These two operators also yield the derivation operators of the context:

$$\mathbb{K}_{A_k}^{i,j} = (K_i, K_j, \{(a_i, a_j) \mid \forall a_k \in A_k,\ (a_i, a_j, a_k) \in Y\}).$$

[5]The plural form of modus is *modi*.

With all these operators, we are finally able to generate a triadic concept from subsets of dimensions. For example, suppose we have $X_1 \subseteq K_1$ and $A_3 \subseteq K_3$, we can generate a triadic concept with the following $Gen(\cdot)$ formula:

$$Gen(X_1, A_3) = (X_1^{(1,2,A_3)(1,2,A_3)}, X_1^{(1,2,A_3)}, (X_1^{(1,2,A_3)(1,2,A_3)} \times X_1^{(1,2,A_3)})^{(3)}).$$

Note that $(X_1^{(1,2,A_3)(1,2,A_3)}, X_1^{(1,2,A_3)})$ is a dyadic concept on the context $\mathbb{K}_{A_3}^{1,2}$. Hence, the general idea of such a generation is to find a dyadic concept on $\mathbb{K}_{A_3}^{1,2}$ with X_1 in its extent and extend it to a triadic concept using the 3-derivation operator. For example, if we start from an object subset $X_1 = \{g_2, g_3\}$ and a condition set $A_3 = \{b_2\}$ from the sample context in Fig. 1.5. We first compute $X_1^{(1,2,A_3)}$, that is, the symptoms that Diseases g_2 and g_3 both have in Stage b_2, which should be $\{m_1, m_4\}$. Then, we compute $X_1^{(1,2,A_3)(1,2,A_3)}$, that is, all diseases having these symptoms in Stage b_1, which should be $\{g_2, g_3\}$. Now we get a dyadic concept $(\{g_2, g_3\}, \{m_1, m_4\})$ on $\mathbb{K}_{A_3}^{1,2}$, which is the dyadic context representing the symptoms of all diseases in Stage b_2. Finally, we are to extend this dyadic concept into a triadic concept by computing $(X_1^{(1,2,A_3)(1,2,A_3)} \times X_1^{(1,2,A_3)})^{(3)}$, that is, the stages during which both Diseases g_2 and g_3 have such symptoms, which would be $\{b_1, b_2\}$. After all these steps, we get the triadic concept represented with the shaded rectangular boxes in Fig. 1.4, that is, $(\{g_2, g_3\}$, $\{m_1, m_4\}$, $\{b_1, b_2\})$.

Unlike the basic FCA case, all triadic concepts extracted from a triadic context CANNOT be organized into a mathematical lattice. Although there have been trials to organize them into other structures called a *tri-lattice* [64], the structure is seldomly studied and discussed in successive research on TCA [48, 57, 85]. Hence, in this thesis, we will not give further introductions to this structure. More details on this structure can be found in [64].

Chapter 2

Fast algorithm for FCA and TCA using zero-suppressed decision diagrams

2.1 Introduction

In this chapter, we propose a fast algorithm called *Z-TCA* for both FCA and its three-dimensional extension, TCA. This algorithm is our solution to the first issue that makes it difficult to work out an FCA4SML method, that is, the lack of a fast algorithm for FCA and its natural extension TCA.

Usually, when we mention "to find a faster algorithm", we mean that to find an algorithm with a lower *time complexity*. However, for FCA, the time complexity of the classic algorithm, *NextClosure*, is already proven to have reached the lower bound $O(|C|)$, where $|C|$ is the number of all concepts. Hence, it is impossible to develop a faster algorithm in terms of time complexity. Similarly, it is proven that the process of TCA can be reduced to two nested FCA processes, meaning that the complexity of TCA is restricted by that of FCA and thus, the complexity of TCA has also reached its lower bound.

Although we cannot reduce the time complexity of FCA and TCA, it is still possible for us to reduce their running time by a constant factor. This is usually achieved by applying special data structures to make the specific operations in the algorithms become faster. One of the most frequently applied data structures is the *binary decision diagram* (BDD) [2]. For example, the *ProperIm* algorithm [84] applies BDD to store and manipulate the formal concepts, which successfully speeds up the process of FCA. The ProperIm algorithm is later modified and adapted to TCA [85], which saves about 25% of the running time compared to the traditional TCA algorithm, *TRIAS* algorithm[48]. The *ZBDD-Growth* algorithm [78] applies *zero-suppressed decision diagram* (ZDD), an improved version of BDD, to FCA. The

algorithm was proven to be one of the fastest FCA algorithms at the time it was proposed [78]. We consider these works interesting but still have further space for improvements for the following two reasons. First, it is proven that ZDDs, as the improved version of BDDs, are proven to be faster in solving combinatorial problems like FCA or TCA, so all algorithms applying BDDs may be further improved by replacing the BDDs with ZDDs; Second, we find a more efficient and stable way to apply ZDDs to FCA or TCA that reduces the average number of nodes of the intermediate ZDDs, making our algorithm able to speed up the TCA process regardless of the features of the input context. We name our algorithm as Z-TCA and implemented several other algorithms in order to compare their efficiency. The comparison group includes the ZDD version of the aforementioned TRIAS-BDD algorithm as well as the TCA-adaption of another ZDD-based improved FCA algorithm, ZBDD-Growth [78]. Here we set up all experiments for TCA because as briefly mentioned above, the TCA process can be decomposed into two nested FCA processes. Hence, the results for TCA also reflect their performances in FCA. The experimental results have shown that when working with a real-world context, our Z-TCA algorithm can speed up the TCA process up to 3 times compared to the baseline TRIAS algorithm and when working with contexts having a density of more than 5%, our algorithm outperforms all other TCA algorithms. Furthermore, our Z-TCA algorithm is the most stable among all three ZDD-based improved algorithms: When the input context is relatively dense, the other two algorithms may become even slower than the baseline TRIAS algorithm, while our algorithm can still maintain its high efficiency in such a case.

This chapter is organized as follows. First in Section 2.2, we are to introduce the preliminaries related to this topic, including the prototype algorithms for FCA and TCA and the basic knowledge on ZDDs. Then in Section 2.3, we are to present our Z-TCA algorithm, and then introduce two TCA-adaptable ZDD-based improved FCA algorithms. In section 2.4, we present the experimental results. Finally in Section 2.5, we draw a conclusion and present our further plans on this topic.

2.2 Preliminaries

2.2.1 Prototype Algorithm for FCA

As introduced in Section 1.2, by using the derivation operators, we can generate a formal concept from any given attribute set. Hence, we can easily derive the following prototype algorithm for FCA.

Algorithm 1 Prototype algorithm for FCA

Input A dyadic context $\mathbb{K} = (G, M, Y)$.
Output A list of dyadic concepts extracted from $\mathbb{K}$.

$L \leftarrow \{(\emptyset^{(1)}, \emptyset^{(1)(2)})\}$
$C \leftarrow \emptyset$
while $L \neq \emptyset$ **do**
    Pick up a concept $(A_1, A_2) \in L$
    $T \leftarrow T \cup \{(A_1, A_2)\}$
    $L \leftarrow L - \{(A_1, A_2)\}$
    for All attributes $m \in M$ **do**
        $(A_1', A_2') \leftarrow ((A_2 \cup \{m\})^{(1)}, (A_2 \cup \{m\})^{(1)(2)})$
        if $(A_1', A_2') \notin T$ **then**
            $T \leftarrow T \cup \{(A_1', A_2')\}$
            $L \leftarrow L \cup \{(A_1', A_2')\}$
        end if
    end for
end while
Output C

It can be easily derived that the process can generate all concepts in the concept lattice $\mathbb{K} = (G, M, Y)$, because for two concepts (A_1, A_2) and (A_3, A_4) satisfying $(A_1, A_2) < (A_3, A_4)$, there must exist an object $m \in A_2 - A_4$ such that $(A_4 \cup \{m\})^{(1)(2)} = A_2$. It can also be easily derived that only formal concepts will be put into the set L, and one concept will only enter the set once. Hence, the complexity of this algorithm should be $O(|M||G| + |M||C|)$, where $|M|$ is the number of attributes, $|G|$ is the number of objects, and $|C|$ is the number of all formal concepts. Since $|C|$ is exponential to $|M|$ and $|N|$, when $|M|$ and $|N|$ approach infinity, $|C|$ should be far larger than $|M|$ and $|N|$, so the complexity can also be considered to be $O(|C|)$, which already reaches the lower bound of such an enumeration process.

In practical, there have been different improved FCA algorithms like *LCM* [105] and *NextClosure* [35]. These algorithms all have the same complexity with the prototype algorithm, but they are expected to be faster in small data for that their complexity have a smaller constant factor.

2.2.2 Prototype algorithm for TCA

Unlike the dyadic case, it is impossible to generate a triadic concept from a subset of a single dimension only. Instead, as the most straightforward idea, we may consider a nested enumeration of $X_1 \subseteq K_1$ and $A_3 \subseteq K_3$ and use the above formula for generating triadic concepts. However, such a process can be simplified for the following two reasons.

- First, we claim that the full enumeration of all $A_3 \subseteq K_3$ is avoidable. Suppose A_3 is not an extent on $\mathbb{K}^{(3)}$. Then, according to the definition of formal concepts, we should have another $B_3 \supset A_3$ such that B_3 is an extent on $\mathbb{K}^{(3)}$. Since $A_3 \subset B_3$, we must have $\mathbb{K}^{1,2}_{A_3} \supseteq \mathbb{K}^{1,2}_{B_3}$ based on the basic property of formal concepts. On the other hand, since B_3 is an extent on $\mathbb{K}^{(3)}$, we must also have $\mathbb{K}^{1,2}_{A_3} \subseteq \mathbb{K}^{1,2}_{B_3}$ because for any $(x_1,x_2,x_3) \in Y$ where $x_3 \in A_3$, we must also have $x_3 \in B_3$, which is the basic feature of an extent. In summary, we must have $\mathbb{K}^{1,2}_{A_3} = \mathbb{K}^{1,2}_{B_3}$. This implies that for any $X_1 \in K_1$, we should have $Gen(A_3,X_1) = Gen(B_3,X_1)$ because clearly that $X_1^{(1,2,A_3)} = X_1^{(1,2,B_3)}$. In other words, any triadic concept generable from A_3 with an arbitrary $X_1 \in K_1$ is also generable from B_3 with the exactly same X_1. Hence, to avoid repetitive generations, it is better to conduct a dyadic FCA process to extract all extents on $\mathbb{K}^3$, and only use these extracted extents for the generation.

- Similarly, given a specific $A_3 \subseteq K_3$, we can show that it is also unnecessary to enumerate all $X_1 \in K_1$. Suppose that $Gen(X_1,A_3) = (C_1,C_2,C_3)$, then, as analyzed above, no matter what $X_1 \subseteq K_1$ is given, (C_1,C_2) should always be a dyadic concept on $\mathbb{K}^{1,2}_{A_3}$, and C_3 can be derived from (C_1,C_2). This implies that instead of enumerating $X_1 \in K_1$ and generating a dyadic concept from X_1, we may consider directly finding out all dyadic concepts on $\mathbb{K}^{1,2}_{A_3}$, that is, to conduct a dyadic FCA process on $\mathbb{K}^{1,2}_{A_3}$.

From the above, the TCA process can be reduced to two nested FCA processes: First, we conduct an outer dyadic FCA process on $\mathbb{K}^{(3)}$. Then, for each dyadic concept (A_3,A_{12}) extracted from $\mathbb{K}^{(3)}$, we conduct an inner FCA process on $\mathbb{K}^{1,2}_{A_3}$ and derive the corresponding triadic concept from each dyadic concept (X_1,X_2) extracted by the current inner FCA process. It can be easily derived that the time complexity of this prototype algorithm is $O(|C| + |T|)$, where $|T|$ is the number of all dyadic concepts on $\mathbb{K}^{(3)}$ and $|C|$ is the number of triadic concepts. Since $|C|$ is exponential to $|T|$, when the number of attributes and objects approaches infinity, the time complexity of the algorithm can also be considered as $O(|C|)$, which also reaches the lower bound of the enumeration process. This idea was first proposed in [57] and is now still considered the most natural working flow for TCA. To the furthest of our knowledge, all later TCA algorithms are based on this prototype algorithm [48, 85]. A sample pseudocode for this prototype algorithm is presented as Algorithm 2 for a clearer view of the process.

Since all TCA algorithms have such a similar working flow, in the rest of the paper, the two nested FCA processes are called *the FCA components* of a TCA process, or the *outer FCA component* and the *inner FCA component*, respectively.

Algorithm 2 Prototype Algorithm for TCA

```
1: Construct 𝕂^(3)
2: for All dyadic concepts (A3, A12) on 𝕂^(3) do
3:     Construct 𝕂^(1,2)_A3
4:     for All dyadic concepts (X1, X2) on 𝕂^(1,2)_A3 do
5:         if (X1, A2)^(3) = A3 then
6:             Output triadic concept (X1, X2, A3)
7:         end if
8:     end for
9: end for
```

2.2.3 ZDD

A *zero-suppressed decision diagram*[1] (ZDD), is a modified version of *binary decision diagram* (BDD), a data structure designed for a compact representation of Boolean variables and expressions [2]. A BDD is defined as a directed acyclic graph with various nodes representing Boolean variables. Each node has two outward edges LO and HI, where the LO edge represents the case when the variable corresponding to the node takes the FALSE value and the HI edge represents the case when it takes the TRUE value. Alongside these variable nodes, there are two terminal nodes representing TRUE and FALSE, of which there are no outward edges. It can be easily derived that any path from the root node to a terminal node represents a Boolean expression. Also, if we regard the truth value of a variable as the existence of an element in a set, a BDD becomes equivalent to a family of sets. For example, given three variables a, b and c, a Boolean expression like $(a \wedge b \wedge \neg c)$ can represent all sets that contain a and b and do not contain c, *i.e.*, {{a,b}}; the expression $b \vee c$ represents all sets that contain either b or c, *i.e.*, $\{\{b\},\{c\},\{b,c\},\{a,b,c\}\}$. Moreover, the basic operations of set families like $A \cup B, A \cap B$, and $A - B$ also have their equivalent Boolean operations $A \vee B, A \wedge B$ and $A \wedge \neg B$. This means that a BDD can be used as a fully functioning and fast implementation of set families thanks to its compact structures [2].

However, when we use BDDs for representing sparse set families, we need a lot of nodes for negative variables, that is, the variables not included in the sets. For example, if we have 4 variables $\{a,b,c,d\}$, to represent a set family $\{\{a\},\{b\},\{c\},\{d\}\}$, we need a long Boolean expression $(a \wedge \neg b \wedge \neg c \wedge \neg d) \vee (\neg a \wedge b \wedge \neg c \wedge \neg d) \vee (\neg a \wedge \neg b \wedge c \wedge \neg d) \vee (\neg a \wedge \neg b \wedge \neg c \wedge d)$ and this results in a relatively complicated BDD, as is shown in the middle of Figure 2.1. To deal with such problems, zero-suppressed decision diagrams (ZDD) are introduced [77]. A

[1] Some publications, especially those early ones, also refer to it as a *zero-suppressed **binary** decision diagram* or a ZBDD [77–79].

ZDD applies a modified rule for BDD that on a path from the root node to a terminal node, any variable not explicitly assigned with a true value will be considered false. With such a rule, The aforementioned set family can be represented in a much more compact structure, as is shown on the right of Figure 2.1. It is proved that when applied for manipulation of set families, ZDDs will have a much higher efficiency compared to the original BDDs, thanks to such a "zero-suppressed" modification [77].

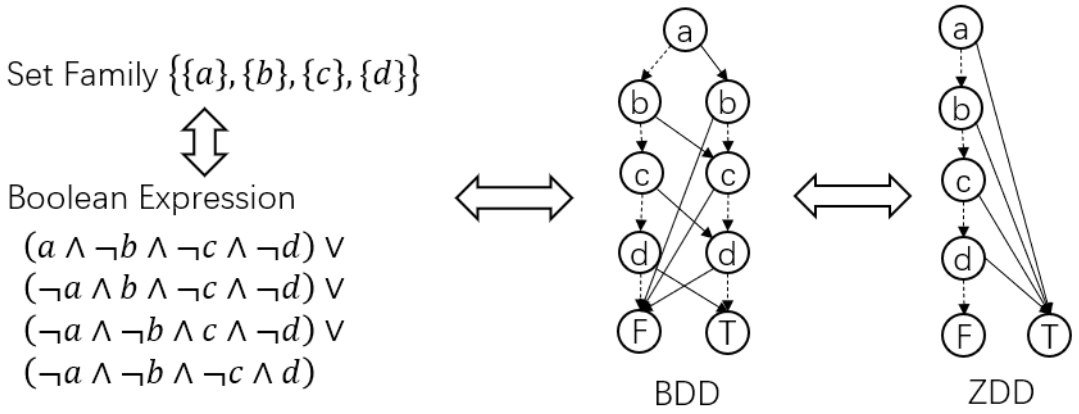


Fig. 2.1 An example of a set family and its equivalent Boolean expression, its BDD representation, and ZDD representation. In the BDD and ZDD representation, the label on each node represents its corresponding variable; the edges in solid lines are the HI edges, and the edges in dashed lines are the LO edge; the node labeled T and F represent the TRUE node, and the FALSE node, respectively.

Both FCA and TCA have a lot of set or set family manipulations, so it is natural to consider applying ZDDs to improve the efficiency of FCA or TCA. There have already been several studies on this topic, which will be explicitly discussed in the next section.

2.3 ZDD-based improved TCA algorithms

In this section, we are to introduce three different ways to apply ZDD for speeding up the TCA process, including our original Z-TCA algorithm as well as two previous TCA-adaptable FCA algorithms – the *ZBDD-Growth* algorithm and the *ProperIm* algorithm. The two previous algorithms are originally proposed for FCA, while they can be easily adapted to TCA following the procedure of the prototype algorithm for TCA introduced in Algorithm 2. Previous research has successfully adapted the ProperIm algorithm to TCA and proved it effective in speeding up the process [85]. However, we are to theoretically analyze that

a direct adaption of these two improved FCA algorithms to TCA must encounter some problems which may slow down the process in some cases. Our specially-designed Z-TCA algorithm, however, is more stable and can keep a high efficiency in any case.

2.3.1 The proposed Z-TCA algorithm

Generally Z-TCA uses a nested *NextClosure* [35] algorithm as the basic framework. The NextClosure algorithm is one of the fastest algorithms for FCA that defines a specific order of all concepts from a concept and enumerates the concepts in that specific order. The algorithm runs in an iteration mode. Suppose the input dyadic context is $\mathbb{K} = (G, M, Y)$, in each iteration, we process a concept (A_1, A_2) and try to generate its next concept in that specific order. To generate the concept, we enumerate all attributes from $m_{|M|}$ to m_1 in the reverse order. Each time for $m_k \notin A_2$, we generate a temporal set A_3 that contains all attributes m_i from A_2 where $i < k$, as well as m_k. We generate a concept $(A_3^{(1)}, A_3^{(1)(2)})$ from this temporal set A_3 by using the derivation operators. If no attribute m_i where $i < k$ and $m_i \notin A_2$ is newly introduced into $A_3^{(1)(2)}$, then the concept $(A_3^{(1)}, A_3^{(1)(2)})$ is thought to be the successor of (A_1, A_2), so now we set the concept in process to $(A_3^{(1)}, A_3^{(1)(2)})$ and start the next iteration. We repeat the process until the concept in process has no successors to be found. More details of this algorithm can be found in the pseudocode shown in Algorithm 3. It is proved that the NextClosure algorithm can enumerate all concepts from the context only once [35]. Hence, its time complexity is linear to the number of all concepts, reaching the lower bound of the complexity of an FCA algorithm. Till now, most TCA algorithms choose to apply the NextClosure algorithm for their FCA components [48, 57, 85].

Although the time complexity of a nested NextClosure algorithm has reached the lower bound, it is clear that we have lots of set operations in the algorithm which can still be further speeded up by implementing them with more efficient data structures like ZDDs, and that would be the core idea of our Z-TCA algorithm. To achieve this, we first change the way we store the formal context by creating in total $|G| + |M|$ ZDDs. Each of the former $|G|$ ZDDs, denoted with LG_k, stores the attribute set of the object $g_k \in G$. Each of the latter $|M|$ ZDDs, denoted with LM_k, stores the set including all objects having attribute $m_k \in M$. Note that our method only uses a ZDD to represent **a single set** instead of a set family. For a set X, we encode it into a ZDD representing the set family $\{\{x\} \mid x \in X\}$. For example, if an object g_1 has attributes m_1, m_2, m_3, then we should have $LG_1 = \{\{m_1\}, \{m_2\}, \{m_3\}\}$. With such a rule, the derivation operators can be easily implemented with ZDDs just as if we are manipulating ordinary sets. The pseudocode for building a context and computing derivation operations in the FCA components of the Z-TCA algorithm are presented in Algorithm 4 and Algorithm 5 respectively.

Algorithm 3 The NextClosure Algorithm

Input A dyadic context $\mathbb{K} = (G, M, Y)$.
Output A list of dyadic concepts generated from $\mathbb{K}$.

for $i \leftarrow 1$ *to* $|M|$ **do**
    Define $p_i \leftarrow \cup_{1 \leq j < i} \{m_j\}$
end for
procedure CLOSURE(M)
    Return $(M^{(1)}, M^{(1)(2)})$
end procedure
procedure FIRSTCONCEPT
    Return CLOSURE($\emptyset$)
end procedure
procedure NEXTCONCEPT((A_1, A_2))
    for $i \leftarrow |M|$ *downto* 1 **do**
        $(X_1, X_2) \leftarrow$ CLOSURE$(A_2 \cap p_i \cup \{m_i\})$
        if $(X_2 \cap p_i = A_2 \cap p_i) \wedge (m_i \in (X_2 - A_2))$ **then**
            Return (X_1, X_2)
        end if
    end for
    Return $(\emptyset, \emptyset)$
end procedure
$(A_1, A_2) \leftarrow$ FIRSTCONCEPT()
while $(A_1, A_2) \neq (\emptyset, \emptyset)$ **do**
    Output (A_1, A_2)
    $(A_1, A_2) \leftarrow$ NEXTCONCEPT(A_1, A_2)
end while

Especially note that the way we apply ZDDs to the storage of the context and the computation of the derivation operators is completely different from the traditional way. Traditionally, a ZDD is used for storing and manipulating a collection of sets, but in our Z-TCA algorithm, we only use them for storing and manipulating a single set. We apply such a different strategy mainly for the following reasons. First, using ZDDs for storing and manipulating sets is still more efficient than traditional set implementations, so our algorithm should be faster than algorithms using a similar nested NextClosure framework but a plain set implementation like TRIAS [48]. This is mainly because ZDDs can share nodes used in previous sets, as is shown in Figure 2.3, which combined with the caching mechanism of ZDDs [77] can speed up the computation of set operations to a large extent. That is, in most cases, we can get the operation results without traversing the whole diagrams because the caching mechanism memorizes the results of previous ZDD operations. For more details about this mechanism, please refer to [77]. Second, our special way of storing the contexts and

computing the derivation operators is more efficient than the traditional way, so our Z-TCA algorithm should also be faster than algorithms using the NextClosure framework but using the traditional way of applying ZDDs like the TCA-adaption of *ProperIm* algorithm [84, 85]. As is shown in Figure 2.2 and 2.4, to compute the derivation operators, our special way of storing and manipulating sets requires exactly the same number of operations compared to the traditional way. However, the sizes of ZDDs using our strategy for storage would be much smaller and as a result, our Z-TCA algorithm would be far faster. More details about the traditional way of storage and manipulation used in the ProperIm algorithm will be introduced later. Third, for algorithms using completely different frameworks for FCA like the ZBDD-Growth algorithm, although it is hard to give a concrete and precise comparison of efficiency, previous studies have shown that in FCA cases, when working with relatively dense contexts, the sizes of intermediate ZDDs used in the ZBDD-Growth algorithm grow sharply, making the algorithm slower than the LCM algorithm [105] using plain set implementations [78]. Since the LCM algorithm has the same complexity as the NextClosure algorithm [35, 105], it is expected that when the density of the context exceeds a certain threshold, the direct TCA-adaption of the ZBDD-Growth algorithm may become extremely slow. Our Z-TCA algorithm, however, does not have such a problem because the sizes of ZDDs used in the algorithm are not related to the number of concepts. Last but not least, the two aforementioned TCA-adaptable ZDD-based FCA algorithms all require extra decoding processes other than the main procedure introduced in Algorithm 2 when adapted to TCA. Our Z-TCA algorithm, however, does not require any extra process. In the next section, we are going to conduct empirical studies to prove that our hypotheses above are correct and that our Z-TCA algorithm indeed has the best overall performance.

Algorithm 4 Build the Context in the FCA components of the Z-TCA algorithm

Input A dyadic context $\mathbb{K} = (G, M, Y)$.

Output A series of ZDDs $\{LG_1, LG_2, \ldots, LG_{|G|}, LM_1, LM_2, \ldots, LM_{|M|}\}$ representing the context.

Note All set families are implemented with ZDDs.

1: Initialize all LG_k, LM_k to $\emptyset$ (Constant False ZDDs).
2: **for** all $(g_i, m_j) \in Y$ **do**
3: $\quad LG_i \leftarrow LG_i \cup \{\{m_j\}\}$
4: $\quad LM_j \leftarrow LM_j \cup \{\{g_i\}\}$
5: **end for**

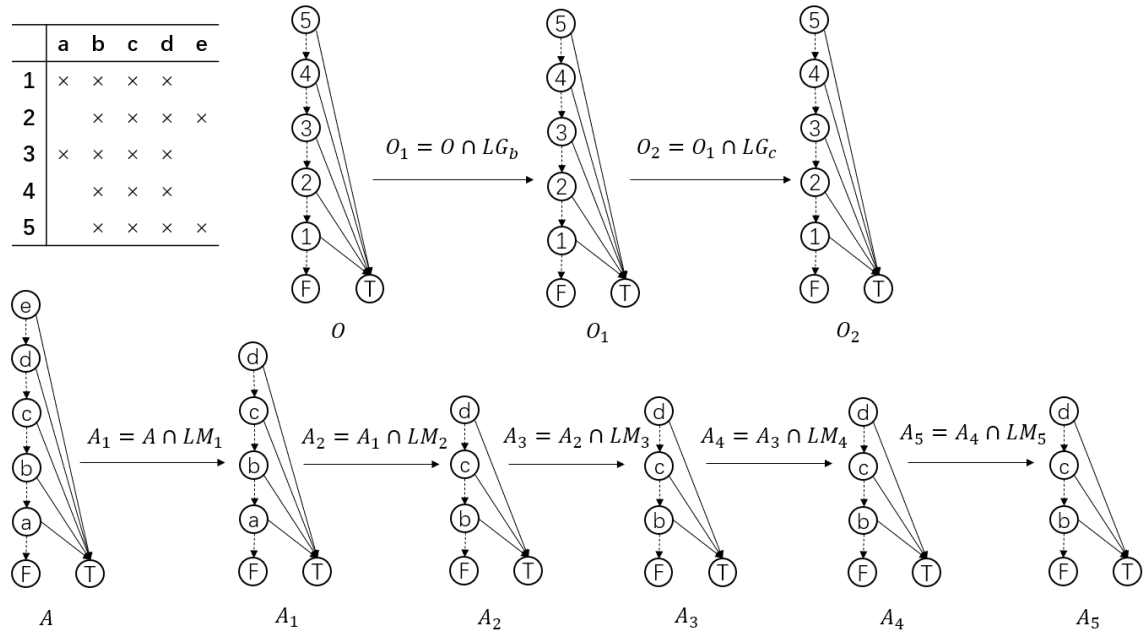


Fig. 2.2 The process for computing $O^{(1)}$ and $O^{(1)(2)}$ where $O = \{b, c\}$ on a sample dyadic context represented with a cross table on the upper-left using the Z-TCA algorithm. The results of $O^{(1)}$ and $O^{(1)(2)}$ are O_2 and A_5, correspondingly.

2.3.2 Previous TCA-adaptable ZDD-based FCA algorithms

ProperIm

This algorithm was first introduced in [84]. It also uses the *NextClosure* algorithm as the framework for FCA and uses BDDs for speeding up the computation of derivation operators. Since a ZDD is an improved version of a BDD, we can easily get a ZDD-adaption of the original ProperIm by replacing all BDDs in the algorithm with ZDDs. In this paper, for convenience, the ZDD-version of the ProperIm algorithm is named with ZDD-ProperIm. In this paper, we only study and discuss the performance of the TCA adaption of ZDD-ProperIm mainly because there is already a previous study on the adaption of the original BDD-version of ProperIm to TCA [85].

The major difference between ZDD-ProperIm and the FCA components of our proposed Z-TCA algorithm is that ZDD-ProperIm applies ZDDs for the storage of contexts and manipulation of sets in a traditional way. In the algorithm, the whole dyadic context $\mathbb{K} = (G, M, Y)$ is collected into one single ZDD, that is, the whole context is converted into a set family F that collects the attribute sets of every object. Given an attribute subset X, the derivation operators $X^{(1)(2)}$ can be easily computed with ZDDs in the following procedure.

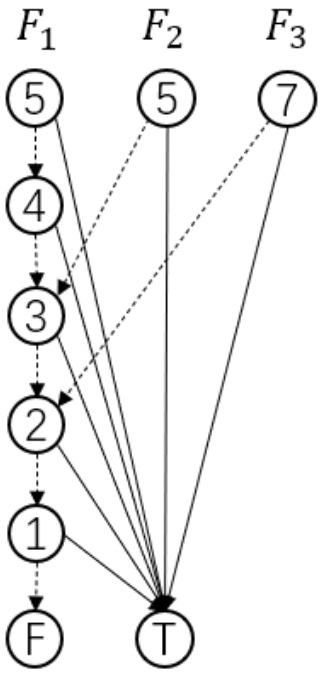


Fig. 2.3 An example of the node usage of three ZDDs: $F_1 = \{\{1\},\{2\},\{3\},\{4\},\{5\}\}, F_2 = \{\{1\},\{2\},\{3\},\{5\}\}, F_3 = \{\{1\},\{2\},\{7\}\}$.

First, we compute $X^{(1)}$ with the "OnSet(F,v)" function which returns the sub-family of F in which all sets include v. By computing the intersection of all OnSet(F,m_1) for all $m_i \in X$ we will get a ZDD R representing the collection of attribute sets of the objects that have all attributes from X. For more details on how the function is implemented with ZDDs, please refer to [77]. Then, we enumerate all $m \in M$ to check if it should be in the intent $X^{(1)(2)}$. If OnSet$(R,m) = R$, then surely that m is included in all sets from R, and thus m should be a member of $X^{(1)(2)}$. The pseudocode for the computations of $X^{(1)(2)}$ is presented in Algorithm 6 and Algorithm 7.

Note that Algorithm 6 only returns R, a ZDD representing the family of all attribute sets of every object in $X^{(1)}$. Although this is enough for the computation of $X^{(1)(2)}$, in case that we want a full list of intents and extents, an extra procedure that converts R into the corresponding object set $X^{(1)}$ is required. The procedure enumerates all objects $g \in G$ and checks whether its attribute set is included in the set family R. The pseudocode for the procedure is presented in Algorithm 8.

As is analyzed previously, the way the ZDD-ProperIm algorithm manipulates sets is less efficient than that used in the FCA components of our Z-TCA algorithm. Moreover, when we want a full list of concepts, we need to spend extra time decoding the ZDDs returned in

Algorithm 5 Compute $B^{(1)}$ and $A^{(2)}$ in the FCA components of Z-TCA algorithm

Input An object subset A or an attribute subset B on a dyadic context $\mathbb{K} = (G, M, Y)$.
Output The results for derivation operators, $Y^{(1)}$ and $X^{(2)}$.
Note All set families are implemented with ZDDs.

1: **procedure** EXTENT((B))
2: $ret \leftarrow \{\{g\} | \ g \in G\}$
3: **for** $m_i \in B$ **do**
4: $ret \leftarrow ret \cap LG_i$
5: **end for**
6: Return *ret*
7: **end procedure**
8: **procedure** INTENT((A))
9: $ret \leftarrow \{\{m\} | \ m \in M\}$
10: **for** $g_i \in A$ **do**
11: $ret \leftarrow ret \cap LM_i$
12: **end for**
13: Return *ret*
14: **end procedure**

Algorithm 6. Our Z-TCA algorithm, however, does not require such an extra process and thus should have better performance.

ZBDD-Growth

The ZBDD-Growth algorithm was first proposed in [78]. This algorithm is initially designed for the task *closed item set mining* [90], while it can also be used for FCA since the two tasks are equivalent. To make it consistent with the content of this paper, here we only introduce the algorithm in terms of FCA but not those used in the original publications.

The ZBDD-Growth algorithm uses a different framework compared to the FCA components of our Z-TCA algorithm and the ZDD-ProperIm algorithm. It uses the *FP-Growth algorithm* [42] as the framework for FCA. In the FP-Growth algorithm, we are to generate all subsets of the attribute set satisfied by at least one object and check whether each is an intent of a concept. The algorithm runs in a recursive depth-first search procedure. Each recursive call is denoted as GROWTH(F), where the parameter F is a set family. The algorithm starts from calling GROWTH(A) where $A = \{S_i\}$ for $i = \{1, 2, \ldots, |G|\}$ and $S_i = \{m \mid (g_i, m) \in Y\}$ for $g_i \in G$. That is, we start by processing the collection of attribute sets of every object. In each recursive call GROWTH(F), we first choose an attribute k to divide F into two groups – the group that all sets contain attribute k, denoted as F_1 and the group that no set contains k, denoted as F_0. Then, we generate a set $F_1' = \{S - \{k\} \mid S \in F_1\}$ and recursively

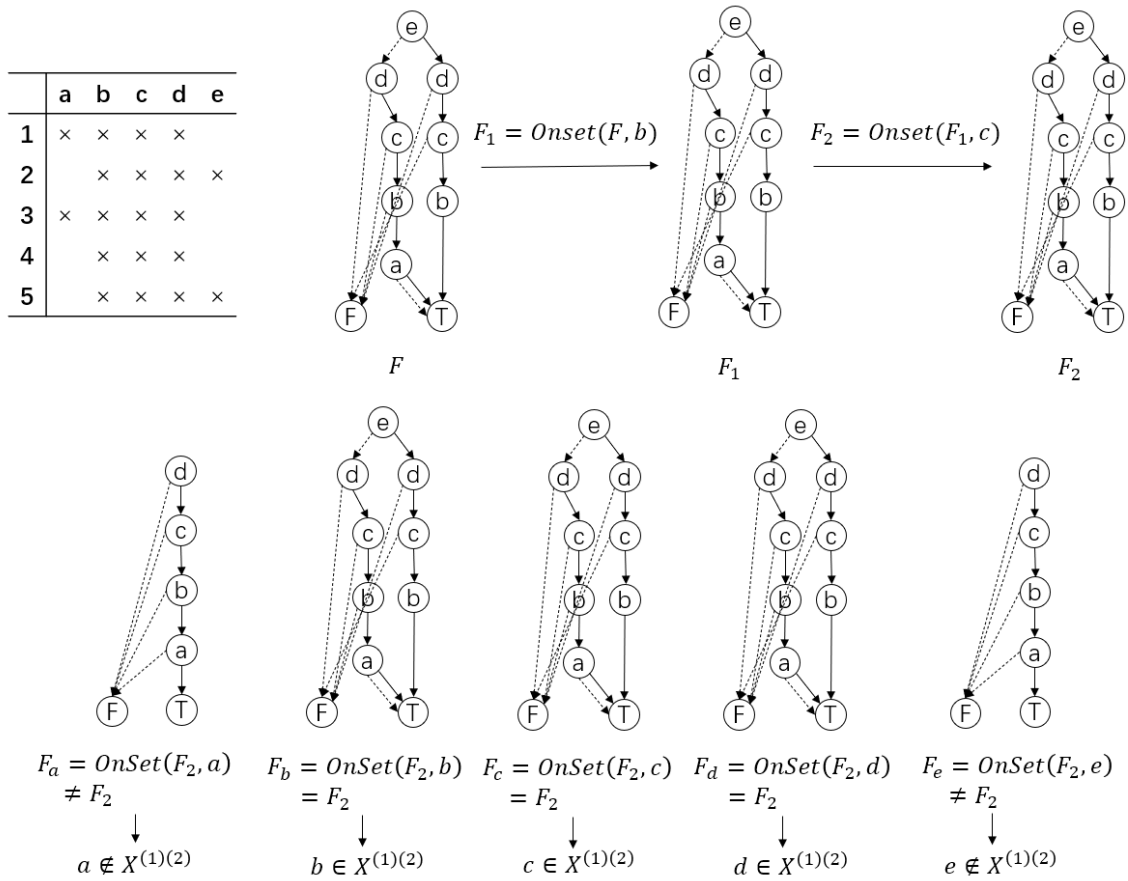


Fig. 2.4 The process for computing $X^{(1)(2)}$ where $X = \{b, c\}$ on a sample dyadic context represented with a cross table on the upper-left using the ZDD-ProperIm algorithm.

call GROWTH(F_1') and GROWTH$(F_1' \cup F_0)$ and collect the results as R_1 and R_0. Clearly, for all $S \in R_1$, $S \cup \{k\}$ is an intent of a concept under the current status, *i.e.*, in the context represented by the current set family F. For a set $S \in R_0$, when it also satisfies $S \in R_1$ and $S \notin S_0$ for all $S_0 \in F_0$, that is, the set S is the subset of some sets including k but is never a subset of any set excluding k, it should never be an intent of concepts as it implies that any object having all attributes from S should also have the attribute k. Hence, it should be pruned out from the collection of intents of concepts, and we should only return $\{S \cup \{k\} \mid S \in R_1\} \cup (R_0 - \{S \mid S \in R_1 \textit{ and } \forall S_0 \in F_0\ S \notin S_0\})$ as the result for the current call.

It can be easily discovered that the FP-Growth algorithm also has a lot of set family parameters and operations. By implementing them with ZDDs, we will get the ZBDD-Growth algorithm. The pseudocode of the recursive procedure of the ZBDD-Growth algorithm is presented in Algorithm 9. Note that in the pseudocode, "TOP(F)" returns the vari-

Algorithm 6 The computation of $X^{(1)}$ in ZDD-ProperIm

Input X, an attribute subset of a dyadic context $\mathbb{K} = (G, M, Y)$.
Output R, the collection of attribute sets of every object in $X^{(1)}$.
Note All set families are implemented with ZDDs.

```
1: procedure EXTENT(F, X)
2:     R ← F
3:     for all m ∈ X do
4:         R ← ONSET(R, m)
5:     end for
6:     Return R
7: end procedure
```

Algorithm 7 The computation of $X^{(2)}$ in ZDD-ProperIm

Input R, the collection of attribute sets of every object in an object set X.
Output The attribute set $X^{(2)}$.
Note All set families are implemented with ZDDs.

```
1: procedure INTENT(R)
2:     ret ← ∅
3:     for all m ∈ M do
4:         if ONSET(R, m) = R then
5:             ret ← ret ∪ {m}
6:         end if
7:     end for
8:     Return ret
9: end procedure
```

able corresponding to the root node of F. "FACTOR0(F, v)" and "FACTOR1(F, v)" return $\{S_0 \mid S_0 \in F \text{ and } v \notin S_0\}$ and $\{S_1 - \{v\} \mid S_1 \in F \text{ and } v \in S_1\}$ respectively. In other words, the former function returns a ZDD representing the collection of all sets from F that exclude v, and the latter one returns a ZDD representing the collection of all sets from F that includes v but with the v removed. "P.PERMIT(Q)" returns a sub-family of P such that for all $S \in P$ there exists an $S_1 \in Q$ such that $S_0 \subseteq S_1$. For more details on how these operations are implemented with ZDDs, please refer to [77]. Also note that since we may encounter the same F multiple times during the recursive search procedure, a cache is used for storing the results for past calls so that we can obtain the results immediately when a set family is processed the second time. The cache can be easily implemented with a hash table [78]. Note that in the hash table, we only need to memorize the pointers of the root nodes of ZDDs instead of the whole diagrams because a reduced and ordered ZDD is canonical for a

Algorithm 8 Expand the Object Set in ZDD-ProperIm

Input R, the collection of attribute sets of every object in an object set X on a dyadic context $\mathbb{K} = (G, M, Y)$.

Output The object set X.

Note All set families are implemented with ZDDs.

```
1: procedure RTOOBJ(R)
2:     ret ← ∅
3:     for all g ∈ G do
4:         if R ∩ {{m | (g,m) ∈ Y}} ≠ ∅ then
5:             ret ← ret ∪ {g}
6:         end if
7:     end for
8:     Return ret
9: end procedure
```

particular set family and variable order. That is, two ZDDs sharing the exactly same root node must be equivalent [77].

Note that after processing, all intents of the concepts are stored within a ZDD. In case we want a list of all concepts, we still need to expand all intents stored in the ZDD and compute their extents. The pseudocode for such a post-process is listed in Algorithm 10. When we use the ZBDD-Growth algorithm as the FCA components for TCA, this post-process is always required simply because the input context of the inner FCA component is built from the intents of dyadic concepts extracted in the outer FCA, which surely costs extra time. Furthermore, the ZBDD-Growth algorithm removes only one attribute from the context at each recursive call, implying that when the context is relatively dense, the algorithm's efficiency will drop sharply. Our Z-TCA algorithm, however, does not have such problems and thus is supposed to have a better overall performance than the direct TCA-adaption of the ZBDD-growth algorithm.

2.4 The experiments

In this section, we are to present the experimental results of our empirical studies on the efficiency of different TCA algorithms. The algorithms we study include the baseline TRIAS algorithm [48], the TCA-adaption of ProperIm algorithm [84] *a.k.a.* the ZDD version of TRIAS-BDD algorithm [85], the TCA-adaption of ZBDD-Growth algorithm [78] and our proposed Z-TCA algorithm. All algorithms are implemented in C++ language. For the algorithms using traditional sets, the sets are implemented with the set container provided by C++ Standard Template Library (STL). For the three ZDD-based algorithms, the ZDDs

Algorithm 9 The ZBDD-Growth Algorithm

Input A, a ZDD representing the collection of attribute sets of every object from a dyadic context $\mathbb{K} = (G, M, Y)$.

Output R, the result ZDD that collects all intents of concepts from $\mathbb{K}$.

Note All set families are implemented with ZDDs.

```
1: procedure GROWTH(F)
2:     if F = ∅ or F = {∅} then
3:         Return F
4:     end if
5:     if there is only one attribute v in F then
6:         Return {v}
7:     end if
8:     if an entry (F,R) is found in the cache then
9:         Return R
10:    end if
11:    v ← TOP(F)
12:    F0 = FACTOR0(F,v)
13:    F1 = FACTOR1(F,v)
14:    R1 = GROWTH(F0)
15:    R0 = GROWTH(F0 ∪ F1)
16:    R ← {S1 ∪ {k} | S1 ∈ R1} ∪ (R0 − (R1 − R1.PERMIT(F0)))
17:    Push (F,R) into the cache
18:    Return R
19: end procedure
20: GROWTH(A)
```

are implemented with SAPPOROBDD, the library developed by the author of [77]. The algorithms are executed on an Ubuntu 18.04 system with a 2.10GHz CPU and 93GB RAM. In the experiments, all algorithms are required to output a full list of triadic concepts, including their extents, intents, and modi. In the rest of this section, when we mention "ZDD-ProperIm" or "ZDD-Growth", we always refer to their TCA-adapted versions.

2.4.1 The experiment on real-world contexts

We first conduct an experiment on a real-world context. The context is built from the Internet Movie Database (IMDb) [2]. IMDb is an online database of information related to movies, dramas, video games, etc. From the database, we extract three groups of variables – the movie staffs, the genres, and the decades as the object set G, the attribute set M, and the condition set B, respectively. Then, if we find a staff g starred in or participated in the production

[2]Official site: https://www.imdb.com/

Algorithm 10 Expand And Output All Concepts From the Result ZDD.

Input R, the result ZDD that collects all intents of concepts from $\mathbb{K} = (G, M, Y)$.

Output The expanded list of dyadic concepts collected in R.

Note All set families are implemented with ZDDs. A(o) represents the attribute set of object o.

```
1: procedure PRINT(F, T, S)
2:     if F = ∅ then
3:         Return
4:     end if
5:     if F = {∅} then
6:         E ← ∅
7:         for each object o ∈ M do
8:             if T ∩ A(o) = A(o) then
9:                 E ← E ∪ {o}
10:            end if
11:        end for
12:        Output (E, S)
13:    end if
14:    v ← TOP(F)
15:    F_0 = FACTOR0(F, v)
16:    F_1 = FACTOR1(F, v)
17:    PRINT(F_1, {S | S ∈ T and v ∈ S}, S ∪ {v})
18:    PRINT(F_0, T, S)
19: end procedure
20: PRINT(R, R, ∅)
```

of a movie categorized with genre m in the decade b, we add a ternary relation (g, m, b) to the collection Y. For example, $(WilliamK.L.Dickson, Documentary, 1890)$ means that William K.L. Dickson participated in the production of a documentary released in the 1890s. Note that all staff is identified with unique IDs in the database, so there is no need to be concerned about distinguishing staff with the same names. After such a process, we get a triadic context with the size $4,153,510 \times 29 \times 17$, of which there are $11,057,177$ incidences. The context has a very similar feature to the previously studied ones – it is sparse, and the object set is much larger than the attribute and condition set. According to previous studies, it may take longer than two weeks to conduct TCA on such a large context using the TRIAS algorithm [48, 85]. Hence, in this experiment, we take five small sections to build five small contexts as five test cases. The features of these test cases as well as the experimental results for running the four algorithms are listed in Table 2.1.

From the results, we can see that in this experiment, the ZBDD-Growth algorithm performs the best, which can run up to 8 times as fast as the TRIAS algorithm in the 5th

Context Size	Density	# of concepts	TRIAS	Z-ProperIm	Z-Growth	Z-TCA
$71 \times 29 \times 17$	0.3%	69	1.0	1.0	0.1	0.3
$188 \times 29 \times 17$	0.5%	325	4.6	4.7	0.6	1.6
$331 \times 29 \times 17$	0.5%	577	9.0	9.0	1.2	3.2
$646 \times 29 \times 17$	0.5%	961	18.8	17.4	2.6	6.7
$1323 \times 29 \times 17$	0.5%	2084	71.0	63.8	9.4	26.6

Table 2.1 Running time of the algorithms on real-world contexts. All times are shown in seconds. Z-ProperIm is for ZDD-ProperIm, Z-Growth is for ZBDD-Growth.

Density/%	0.5	1.5	2.5	3.5	4.5	5.5	6.5	7.5	...	17.5
TRIAS	3.8	9.2	14.8	24.6	42.5	63.0	95.7	136.6	...	2451.7
ZDD-ProperIm	4.2	9.8	15.9	27.3	50.8	77.2	127.0	198.3	...	13653.2
ZBDD-Growth	**0.9**	**3.1**	**5.5**	**9.1**	**16.8**	26.1	46.7	76.3	...	3262.0
Z-TCA	1.2	3.3	5.5	9.3	17.0	**25.8**	**40.0**	**57.6**	...	**1180.1**

Table 2.2 Running time of the algorithms on randomly-generated contexts with the same size $100 \times 30 \times 20$ but different densities. All times are shown in seconds. Data shown in bold fonts represent the best result for each test case.

case. Our Z-TCA algorithm can also save more than half of the running time compared to the TRIAS algorithm. ZDD-ProperIm can speed up the process, but can only save about 10% of the running time. Since the density of the whole context is almost the same as all five sections, we infer that when we conduct TCA on the full context, the results will be similar to those of these five sections.

Although our Z-TCA algorithm is not the fastest choice in this experiment, its effect of saving the running time is still significant. Also, we analyze that it is the extreme sparsity of this data set that contributes to such a good performance of the ZBDD-Growth algorithm. As the density of the database increases, the sizes of the ZDDs in the ZBDD-Growth algorithm as well as the ZDD-ProperIm algorithm will rise sharply, which may cause a sharp increase in the running time. Our Z-TCA algorithm, however, can suppress the over-growth of the size of the ZDDs because we only use a ZDD for storing a single set instead of a set family. In the next subsection, we will conduct additional experiments to show that our hypothesis is correct.

2.4.2 The experiment on contexts of different sizes and densities

Although real-world triadic contexts are often very sparse [85], we should never assume all contexts to have a density of only 0.5% like the one we used in the previous experiment. To give a more comprehensive study on the performance of these algorithms, it is still necessary

to conduct experiments on contexts of different sizes and/or densities. Since it is hard to find a group of real-world contexts with arbitrary size and density, we are to generate the contexts randomly in this experiment. For relatively dense contexts, we use the SCGaz [94] tool for the generation. For contexts with a density below the minimum required density[3] of the SCGaz tool, we are using the following procedure for the generation. Suppose the context to be generated is $\mathbb{K} = (K_1, K_2, K_3, Y)$, first, we determine the size of the context, *i.e.*, the size of K_1, K_2 and K_3 respectively. Then, for each $k_1 \in K_1$, $k_2 \in K_2$, and $k_3 \in K_3$, we generate a random real number between 0 and 1. If the random number is smaller than the pre-determined density ρ, we add (k_1, k_2, k_3) to Y. In other words, every ternary relation on $K_1 \times K_2 \times K_3$ is generated independently with the probability ρ.

We first study the performance of our algorithms on contexts of different densities. We generate eight contexts with the same size but different levels of densities as the test cases. The size of the contexts is set to $100 \times 30 \times 20$, which is about the same as the size of test case #1 in the last experiment. A part of the results is collected in Table 2.2. From the results, we can see that when the density of the context reaches about 5%, our Z-TCA algorithm becomes faster than the ZBDD-Growth algorithm. Moreover, as the context becomes denser, the running time of both ZDD-ProperIm and ZBDD-Growth becomes closer to the baseline TRIAS algorithm; when the density reaches 17.5%, both algorithms fail to speed up the process and take more time than the baseline. This perfectly matches our hypothesis. Hence, we can conclude that if the input context is not extremely sparse, our Z-TCA algorithm should be the best choice for TCA among all these four algorithms. Also, our algorithm is the most stable among all three ZDD-based improved algorithms, as it can always reduce the running time regardless of the features of the input context.

To further study the density threshold above which our Z-TCA algorithm can outperform the ZBDD-Growth algorithm, we generated several other contexts of different sizes to test the two algorithms. Part of the results is shown in Table 2.3. It is clear from the results that the density threshold is around 4.5% to 5.5%, which is almost independent of the context size. That is, no matter what size the input context has, in most cases, as long as the density of the context exceeds around 5%, our Z-TCA algorithm will run faster than the ZBDD-Growth algorithm and become the best choice for TCA.

Finally, as analyzed in Section 3.1, the caching mechanism for ZDDs contributes to the high efficiency of our algorithm. Hence, to further study how much it influences efficiency, we implemented another version of the Z-TCA algorithm with the caching mechanism for ZDDs disabled. We compare the efficiency of this modified Z-TCA algorithm to the original

[3]The SCGaz tool is used for generating dyadic contexts without redundant objects, *i.e.*, objects having the same attributes. Clearly, if a context is too sparse, there should always be some redundancy.

Context Size	Density	ZBDD-Growth	Z-TCA
$30 \times 30 \times 30$	4.5%	**5.5**	6.7
$30 \times 30 \times 30$	5.5%	10.4	**10.0**
$100 \times 30 \times 30$	4.5%	**54.1**	57.6
$100 \times 30 \times 30$	5.5%	96.1	**90.3**
$200 \times 30 \times 10$	4.5%	6.1	**5.1**
$200 \times 30 \times 10$	5.5%	10.3	**8.9**
$200 \times 30 \times 20$	4.5%	**52.9**	53.8
$200 \times 30 \times 20$	5.5%	94.4	**90.4**
$200 \times 40 \times 20$	4.5%	**128.1**	130.0
$200 \times 40 \times 20$	5.5%	225.3	**222.2**

Table 2.3 Running time of the algorithms on random contexts with different sizes and the density around the borderline. All times are shown in seconds. Data shown in bold fonts represent the better result for each test case.

Context Size	Z-TCA	Z-TCA (No Caching)	TRIAS
$30 \times 30 \times 30$	7.3	25.1	20.9
$200 \times 30 \times 10$	5.0	14.5	11.2
$100 \times 30 \times 30$	58.3	182.5	168.1
$200 \times 30 \times 20$	50.9	152.9	127.6
$200 \times 40 \times 20$	131.7	406.8	350.9
$300 \times 40 \times 20$	286.9	892.9	767.2

Table 2.4 Comparison of the running time of our Z-TCA algorithm with or without the caching mechanism for ZDDs.

version and the baseline TRIAS algorithm by testing them on another group of randomly generated contexts with different sizes and a density of 4.5%. The results are collected in Table 2.4. From the results, it is clear that with the mechanism removed, our Z-TCA algorithm becomes slower than the baseline TRIAS algorithm. Our analysis is that without the caching mechanism, one will have to traverse from the root node to a terminal node when conducting ZDD operations. In such a case, the ZDDs used in the algorithm degenerate into linked lists. Since our Z-TCA algorithm has the same overall flow as the TRIAS algorithm and the only difference between the two algorithms is the set implementation, it is clear that the Z-TCA algorithm without the caching mechanism should be slower than the TRIAS algorithm because linked lists are less efficient than red-black trees used in the C++ standard template library. This matches our assumption and proves that the caching mechanism for ZDDs indeed has a great effect in speeding up the set operations.

2.5 Conclusion of the chapter

We studied three ways of applying ZDDs to speed up the process of TCA – our proposed Z-TCA algorithm, the TCA-adapted ZDD-ProperIm algorithm, and the TCA-adapted ZBDD-Growth algorithm. Unlike the other two algorithms, our Z-TCA algorithm does not use a ZDD for storing a set family but only uses it for representing a single set. Experimental results have shown that in an extremely sparse real-world context, our Z-TCA algorithm can save about 66% of the running time compared to the baseline TRIAS algorithm. Although it is slightly slower than the TCA-adapted ZBDD-Growth algorithm, experimental results of another empirical study have proved that when the density of the context exceeds a threshold, which is about 5%, our Z-TCA algorithm will outperform the TCA-adapted ZBDD-Growth algorithm and become the fastest one among all four algorithms. Moreover, when the density continues to grow, the other two ZDD-based improved algorithms eventually become slower than the baseline TRIAS algorithm. Our Z-TCA algorithm, however, can still save about 50% of the running time, showing high stability and good overall performance. Since TCA is equivalent to two nested FCA processes, our algorithm can also be applied to dyadic FCA tasks. When our method is applied to an FCA4SML method, we will spend less time in the FCA or TCA process and thus have more time in optimizing the predictive model, which enables the method to get better prediction results.

Although our Z-TCA algorithm has improved the efficiency of TCA to a great extent, it is still very difficult to conduct TCA on a full real-world context. Hence, as our future work, we plan to propose a data structure for a more efficient memoization of the intermediate results of set operations, which will hopefully further reduce execution time. Moreover, since the number of triadic concepts extracted from a full-size real-world context may be very large and thus hard for human analysis, we plan to propose a method for abstracting the differences of the objects to reduce the number of concepts and speed up the extraction process for TCA. There has already been related work on concept reduction of the basic FCA [17, 26] (which will be explicitly discussed in the next chapter), while for our future work, we plan to study the adaption of these methods to TCA. By combining these methods with our fast Z-TCA algorithm, we may further improve the practicability of TCA and unleash the full potential of this task.

Chapter 3

Reducing the concept lattice using integer programming

3.1 Introduction

In this chapter, we propose a method for reducing the number of objects and formal concepts using *integer programming* [19]. This is our first solution to the second issue that makes it hard to work out an FCA4SML method, *i.e.*, the number of formal concepts is so large that they are difficult to be further processed by an SML method.

If we want to work out an SML method, this issue becomes more urgent, as an SML method usually need to access the data for multiple epochs in order to fit the optimization problem better, and any little increase to the data will cause a great increase in the running time. The reason why the concept lattice may easily become excessively large is that FCA is too "precise". For example, in the context shown in the left panel of Fig. 1.2 in chapter 1, we can see that Disease g_1 has very similar symptoms to Diseases g_2 and g_3, except that it does not have Symptom m_1. In practical cases, it is sometimes possible to categorize Diseases g_1, g_2 and g_3 into a class, and the symptoms of the class of diseases broadly include Symptoms m_1, m_2 and m_3. However, for FCA, there is no such option because when we apply FCA on the context, we will always get two different concepts $(\{\mathrm{g}_1, \mathrm{g}_2, \mathrm{g}_3\}, \{m_2, m_3\})$ and $(\{\mathrm{g}_2, \mathrm{g}_3\}, \{m_1, m_2, m_3\})$ which clearly differentiate Disease g_1 from Diseases g_2 and g_3. Such cases are extremely common in real-world formal contexts. As a result, FCA might then extract a vast number of complicated concepts representing minor features of the data, causing great difficulty in human analysis and further processing by an RML or SML method [26].

Methods for attacking this problem are categorized as *concept lattice reduction* [26], which aims to approximate the original context into a reduced context with fewer or simpler objects and attributes and corresponds to a concept lattice with fewer concepts. The reduction process, *i.e.*, the generation of the reduced context, should follow the basic principle that it should only "remove excessively detailed difference" but not cause a significant change in the context and the knowledge in the concept lattice [26].

There are many different categories of concept lattice reduction methods [26, 44, 47, 101, 102]. In this research, however, we focus on a specific group of methods based on the *object reduction* ideology, which generates the reduced context by identifying and merging similar objects in the original context. For example, in the context shown in the left panel of Fig. 1.2, since Diseases g_1, g_2 and g_3 all have similar symptoms, we may choose to abstract their minor differences by *merging* them, or namely, to modify the context and make them have exactly the same attributes[1]. After merging, the reduced context and its corresponding concept lattice are shown in Fig. 3.1. As can be seen in the figure, the concepts representing the minor features of Diseases g_1, g_2 and g_3 in the original concept lattice also merge into a generalized concept, which reduces the concept lattice.

While object reduction methods are widely preferred thanks to their simplicity and effectiveness [17, 25, 44, 59], we point out that these may have two problems violating the basic principle of concept lattice reduction. First, most previous methods lack a mechanism for monitoring the overall similarity between the generated approximate context and the original one. As a result, their generated "approximate" context may introduce many modifications to the original context that is hard to be considered a valid "approximation". Second, merging similar objects will not always reduce the concepts representing minor features as expected. It may instead cause other changes to the concept lattice. For example, in the aforementioned sample context in the left panel of Fig. 1.2, if we merge Diseases g_4 and g_5 to make both of them have all symptoms that either Disease g_4 or Disease g_5 has, we will find that no concepts merge. Even worse, a new concept $(\{g_4, g_5\}, \{m_1, m_2, m_3, m_4\})$ which is not a generalization of any concept in the original concept lattice is inserted into the "reduced" concept lattice, making it a failed reduction. The "reduced" context and its concept lattice resulting from this failed reduction are shown in Fig. 3.2.

This example might appear contrived since the symptoms of Diseases g_4 and g_5 are not similar enough, in Section 3.3, we will give a more detailed example showing that merging the most similar objects will also cause the same problem. Similarly, in some other cases, we may find some concepts in the original lattice eliminated after reduction, which will

[1] In a formal context, the only information that differentiate objects is the attributes. If two objects have exactly the same attributes, they will become completely equivalent.

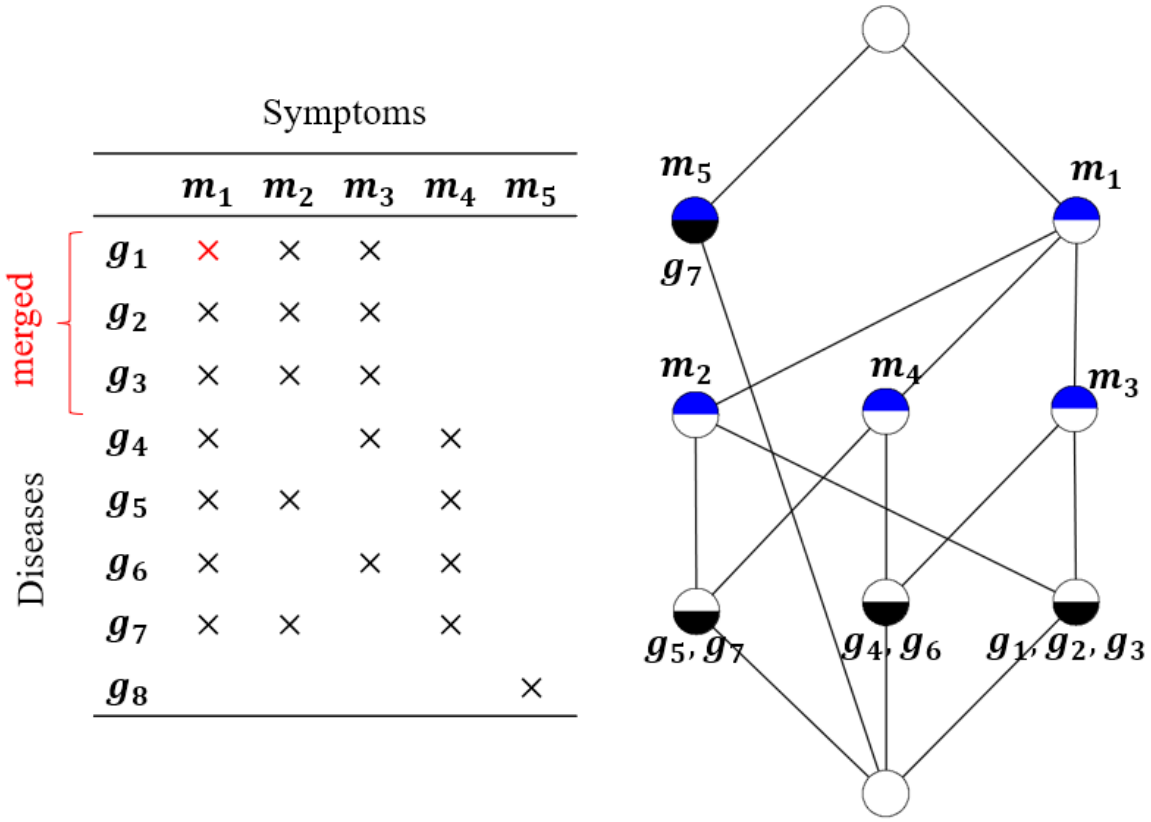


Fig. 3.1 Left: The reduced formal context after merging Diseases g_1, g_2 and g_3 in the context shown in left panel of Fig. 1.2. Right: the corresponding concept lattice of the reduced context.

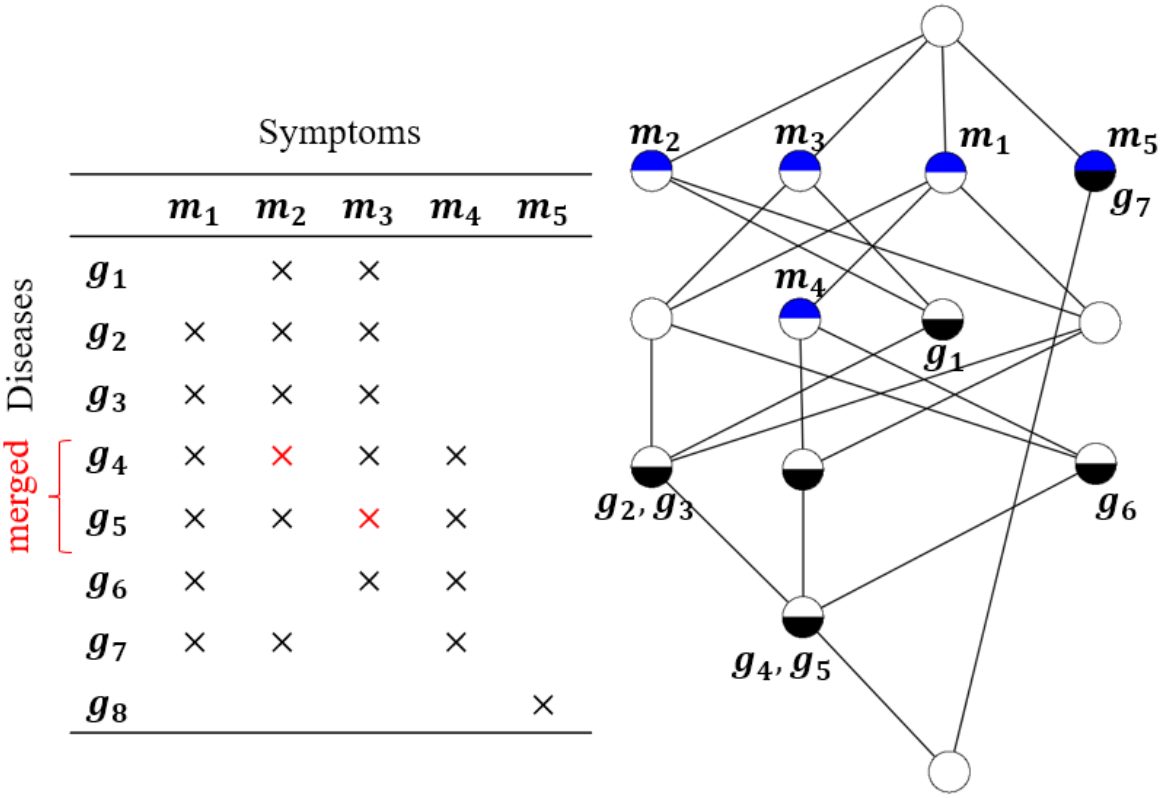


Fig. 3.2 Left: The reduced formal context after merging Diseases g_4 and g_5 in the context shown in left panel of Fig. 1.2. Right: the corresponding concept lattice of the reduced context.

also be explicitly discussed in Section 3.3. We claim that these insertions and eliminations will cause changes to the knowledge in the concept lattices and should be considered as a cost of the reduction. However, no previous method has any mechanism for controlling their volumes or even monitoring their occurrences. As a result, these methods may risk significantly changing the knowledge of the concept lattice.

Aiming to solve both problems, we propose a new concept lattice reduction method using *integer linear programming* (ILP). With ILP, we can determine the optimal values of *integer variables* maximizing a target function under a group of *linear constraints* [19]. We use ILP because, first, by regarding the formal context as a collection of Boolean variables, the aim of object reduction can be naturally translated into a linear optimization problem that ILP can solve. With such a translation, the overall similarity of the approximate context and the original one as well as the number of inserted/eliminated concepts can all be computed with linear operators, as we prove in Section 3.4. Hence, our method can limit the maximum changes caused to the input context and the output concept lattice simply by adding more linear constraints, which solves the two problems pointed out above. We conduct experiments on different randomly-generated contexts and sections of real-world contexts to show that our method can solve the above problems and produce reduction plans that more closely conform to the principle of concept lattice reduction.

Note that in this chapter, we will frequently discuss and compare the formal contexts before and after reduction. To improve the readability, we will use italic letters like T, T' to represent formal contexts instead of the standardized outlined letter $\mathbb{K}$. Similarly, we will use $\mathrm{L}(T)$ for representing the concept lattice of context T instead of the standardized denotation $\underline{\mathfrak{B}}(\cdot)$. Furthermore, since we will frequently discuss the derivation operation of some sets on different contexts, to make it more concise, in this chapter we will use $\mathrm{A}_T(X)$ for representing $X^{(2)}$ on context T, and $\mathrm{O}_T(Y)$ for representing $Y^{(1)}$ on context T.

3.2 Preliminaries

Given a formal context T, *concept lattice reduction* can be roughly defined as the process that generates a *reduced context* T' such that $\mathrm{L}(T')$ is considered *simpler* than $\mathrm{L}(T)$ without significantly changing the information in T. Currently, there are many different standards for what should be called a *reduced context* and which changes should be considered *significant*, so it is hard to give a formal definition for this task [18, 26, 66, 74, 91, 104, 118]. However, there is a specific sub-task of concept lattice reduction, called *object reduction* [26], which has a relatively fixed flow and an objective goal.

Given a formal context $T = (G, M, Y)$, an *object reduction* is a two-step process that generates a *reduced context* $T' = (G, M, Y')$ from the *original context* T. The two steps are as follows.

1. First, the object set G is divided into k disjoint subsets $C = \{C_1, C_2, \ldots, C_k\}$ such that $C_1 \cup C_2 \cup \ldots \cup C_k = G$. Each subset is also called a *similar group*, since objects in the same group should be *similar* to each other.

2. Then, objects classified into the same similar group are *merged*. Namely, these objects are modified to have exactly the same attributes in the reduced context T', following a pre-determined merging rule. The merging rule should either be OR or AND. For any object $g_i \in G$ classified into subset $C_j \in C$, we should have:

   - If the OR rule is applied, it will be modified to have all attributes that **at least one** object in C_j has in the original context T. That is, in this case we should have:

   $$\mathrm{A}_{T'}(g_i) = \bigcup_{g_a \in C_j} \mathrm{A}_T(g_a).$$

   - If the AND rule is applied, it will be modified to have only the attributes that **all** objects in C_j have in the original context T. That is, in this case we should have:

   $$\mathrm{A}_{T'}(g_i) = \bigcap_{g_a \in C_j} \mathrm{A}_T(g_a).$$

In other words, the core idea of object reduction is to *merge groups of similar objects*, and the way it merges a group of similar objects is to modify the context to make all of these have exactly the same attributes.

There are two points needed to be noted about the definition. In the first step, the measure for the *similarity* of objects varies among different methods, and the choice for similarity measure is usually the featured part of an object reduction method. For example, in [25], the authors assigned weights for attributes and thus the similarity of objects can be measured by the difference of their attribute weight sums; in the second step, to the furthest of our knowledge, all previous object reduction methods assume that the merging rule should be applied globally [17, 25, 59, 100]. That is, all similar groups must be merged with the same rule in a single object reduction process.

After generating the reduced context T' from T, we need to evaluate the quality of the reduction. In previous research, a reduction is thought to have high quality if both of the subsequent metrics are high [17, 25, 59].

- **The number of objects reduced**, which is defined to be $|G| - l$, where l is the number of "unique" objects in the reduced context T'. If two or more objects have exactly the same attributes, these are only counted as one unique object because these are equivalent.

- **The number of concepts reduced**, which is defined to be $|\mathrm{L}(T)| - |\mathrm{L}(T')|$, where $|\mathrm{L}(T)|$ means the number of concepts in the concept lattice corresponding to context T.

We think these metrics are not enough to give a thorough evaluation of the quality of a reduction because these only reflect the *effectiveness*, but not the *cost* of the reduction. That is, even if some reduction has achieved reducing a lot of objects and concepts, it is still of low quality if the reduction has modified a lot of incidences in T, or the knowledge represented by the concepts in $\mathrm{L}(T')$ is significantly inconsistent with that in $\mathrm{L}(T)$. If we take such costs of reductions into consideration, we will find that most previous methods carry the risk of modifying too many incidences or changing too much information in the concept lattice, as mentioned in the previous section. We will explicitly discuss these problems in the next section.

3.3 Problems of previous object reduction methods

3.3.1 Excessive modification of the context

It can be easily derived from the definition that an object reduction does not add or remove objects or attributes, but only modifies incidences in the context. Hence, suppose an object reduction reduces context $T = (G, M, Y)$ to $T' = (G, M, Y')$, *the number of incidences modified* in the reduction, which equals $|(Y - Y') \cup (Y' - Y)|$, should be a reasonable metric for measuring what extent changes have been caused to the context by the reduction.

With such a definition, we can find a typical problem in previous methods. That is, to reduce a certain number of objects and concepts, the reductions generated by previous methods usually modify many more incidences than the optimal reduction. For example, suppose that now we are to reduce a sample context shown in the left panel of Fig. 3.3, whose corresponding concept lattice is shown in the right panel of the figure. The method proposed in [17] will choose to merge the object group $\{g_1, g_2, g_3, g_4\}$, which modifies 15 incidences but only reduces 3 objects and 4 concepts[2]. However, by merging object group $\{g_2, g_3\}$ and $\{g_5, g_6, g_7\}$, we can reduce 3 objects and 6 concepts at a far lower cost by modifying only 5

[2]The example and reduction plan are retrieved from [17].

incidences. The two reduced contexts and their corresponding concept lattices are shown in Fig. 3.4.

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9
g_1						×			×
g_2	×	×			×				
g_3		×			×			×	
g_4	×			×		×		×	×
g_5	×						×		
g_6			×				×		
g_7	×		×						

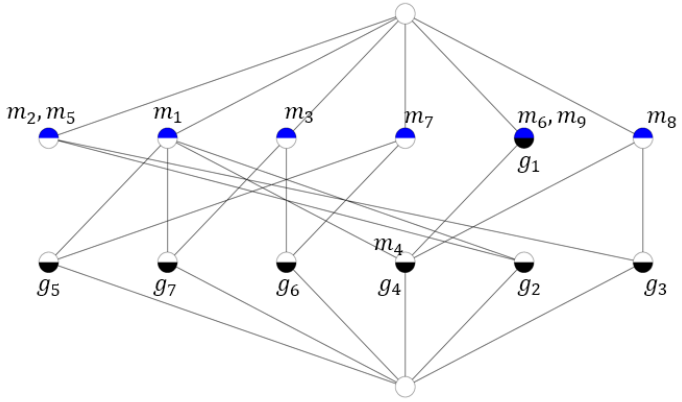


Fig. 3.3 Left: A sample context. Right: The concept lattice corresponding to the context in the left panel.

The cause of this problem is simple. For most previous object reduction methods, any two objects will be merged as long as their similarity exceeds a certain threshold [17, 59], which carries a high risk of merging dissimilar objects since "similarity" has no transitive property.

3.3.2 Unexpected insertion and elimination of concepts

As originally mentioned in [17], merging similar objects with the OR rule will generalize them and remove their minor differences, which will also cause the concepts representing their minor differences to merge (described as "collapse" in the original publication) into a large and generalized concept [17, 26]. However, this might not always be the case. For example, in the context in the upper left of Fig. 3.5 clearly shows that g_1 and g_2 are of the most similar object pairs, but if we merge them with the OR rule, we will find that although some concepts are generalized, none of them merge after reduction. Even worse, two extra concepts that are not a generalization of any concept in the original concept lattice are inserted into the "reduced" lattice, making it a completely failed reduction. The "reduced" context and its corresponding lattice are shown in the lower row of Fig. 3.5. This problem was also mentioned and discussed in [58]. Similar problems might also happen if we apply the AND rule. For example, in the context shown in the upper left of Fig. 3.6 clearly shows that g_3 is very similar to g_1 and g_2, while g_4 is very similar to g_5. If we merge g_3 together with g_1 and g_2 and merge g_4 together with g_5 applying the AND rule, as shown in the lower

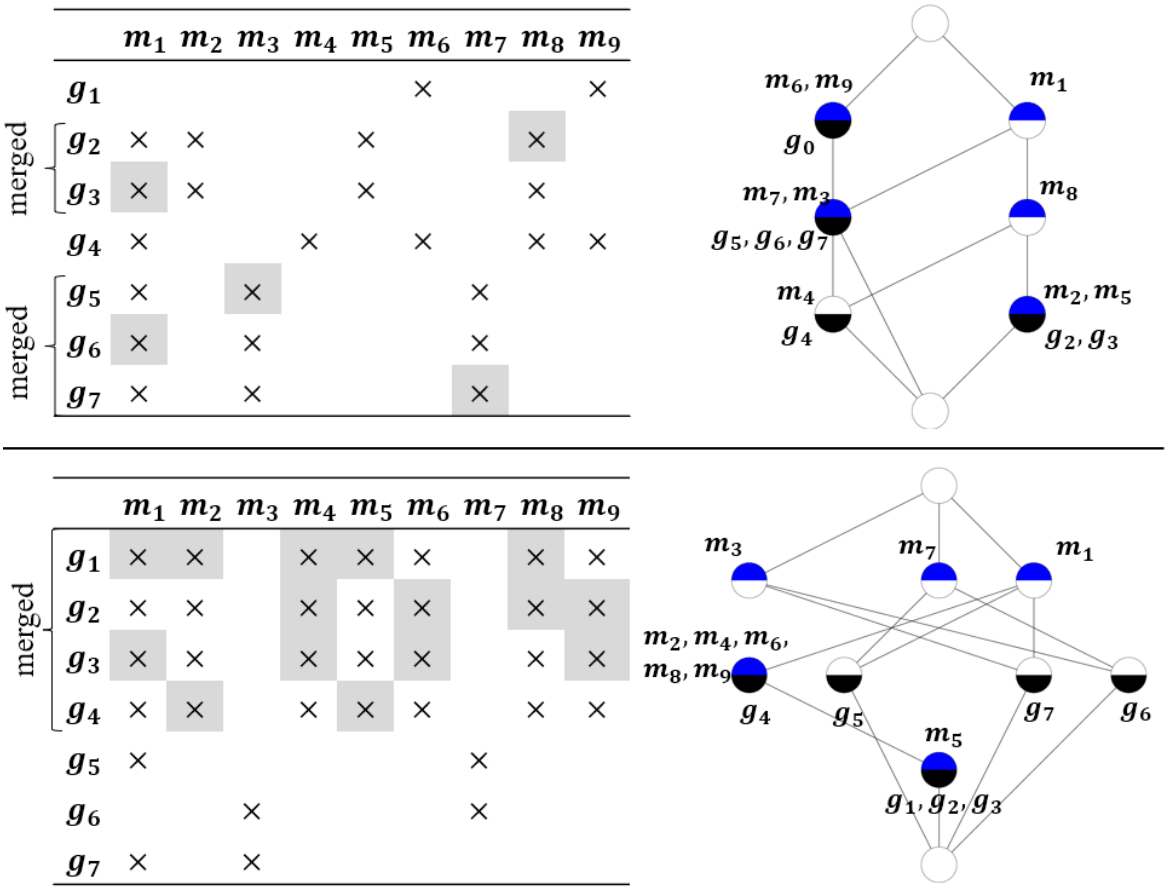


Fig. 3.4 Upper row: a sample reduction plan for the context shown in the left panel of Fig. 3.3. Lower row: the reduction plan given by the method proposed in [17]. Shaded cells represent the modified incidences.

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8
g_1		×	×	×	×			
g_2	×		×	×	×			
g_3	×	×				×		
g_4		×						
g_5	×					×		
g_6		×	×	×	×	×	×	×
g_7	×		×	×	×		×	×

⇩

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8
g_1	×	×	×	×	×			
g_2	×	×	×	×	×			
g_3	×	×				×		
g_4		×						
g_5	×					×		
g_6		×	×	×	×	×	×	×
g_7	×		×	×	×		×	×

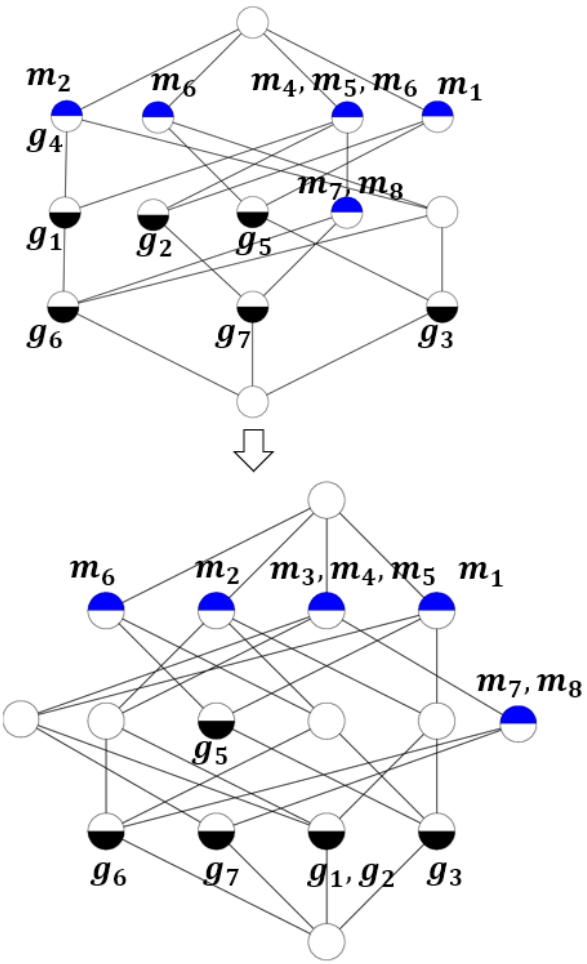


Fig. 3.5 Upper row: a sample context and its corresponding concept lattice. Lower row: the reduced context after merging object g_1 and g_2 with the OR rule, with its corresponding concept lattice shown on the right.

row of Fig. 3.6, we will find that among all seven concepts in the original concept lattice, four of them are specialized but do not merge with each other, while the three concepts completely disappear.

When an object reduction method is applied to a real-world knowledge discovery task, we assert that such insertions and eliminations of concepts may cause unacceptable changes to the knowledge in the concept lattice. For example, if we regard the objects and attributes in the aforementioned contexts as diseases and symptoms, correspondingly. Then, by applying object reductions with either the OR rule or the AND rule, we may expect to get generalized or specialized classes of diseases, correspondingly. However, the insertion of two extra concepts in the example shown in Fig. 3.5 would mean that the reduction process has **created a new class of diseases**, and the elimination of the concept in the example shown in Fig. 3.6

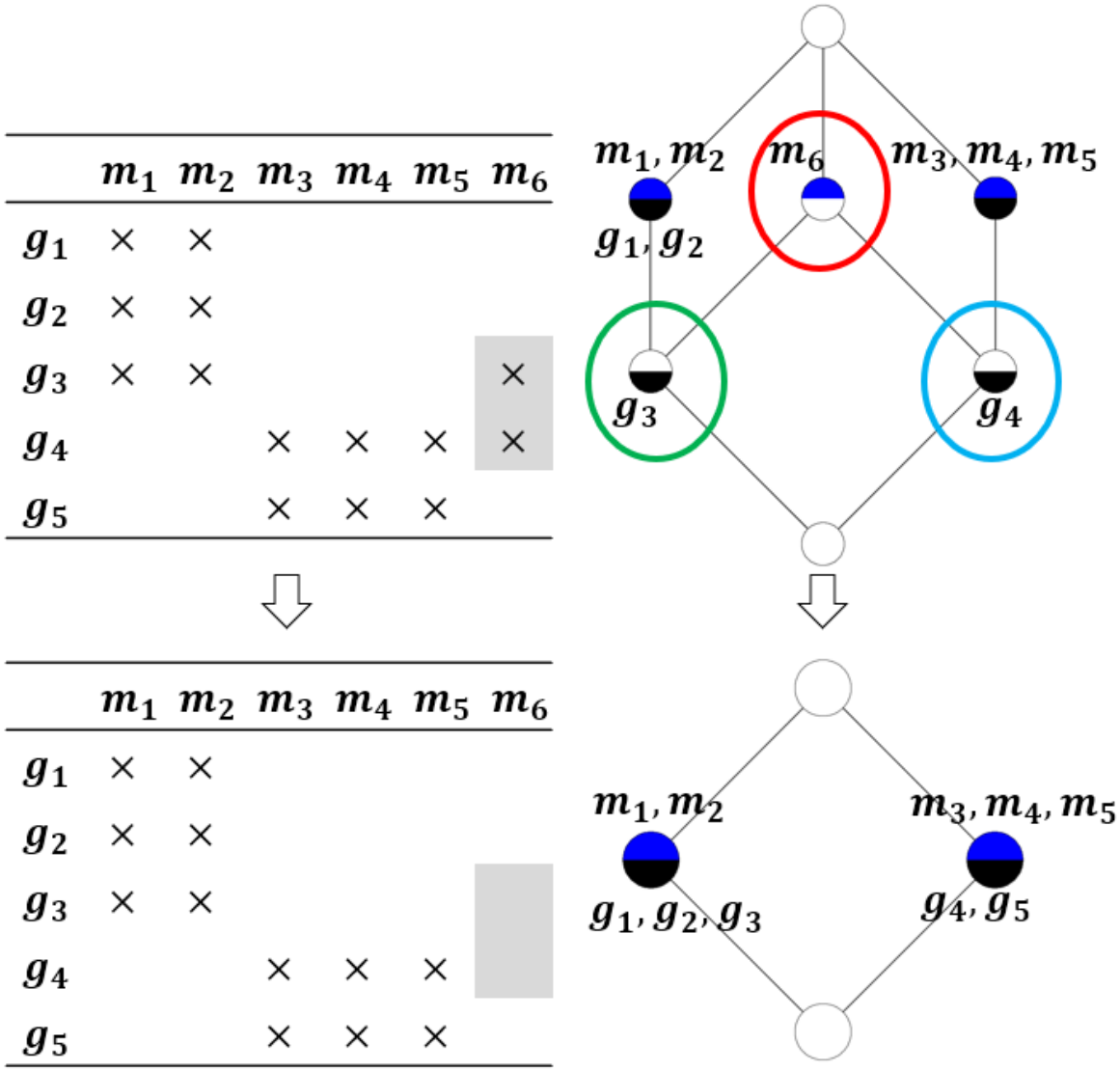


Fig. 3.6 Upper row: a sample context and its corresponding concept lattice. Lower row: the reduced context after merging g_3 with g_1 and g_2, and merging g_4 with g_5 applying the AND rule, with its corresponding concept lattice shown on the right.

would mean that the reduction process has **removed the information of some classes of diseases**. Clearly, if such cases frequently occur, it will violate the concept lattice reduction principle that aims to minimize the changes to the knowledge of the original concept lattice.

There have been studies on quantifying such changes to the concept lattice [27, 65]. To the furthest of our knowledge, most previous methods quantify the changes at the object/attribute level. That is, these method study the degree of changes by analyzing how many implication rules of objects/attributes are inserted or eliminated after reduction [27]. In this research, however, we assert that quantifying the degree of inconsistency at the concept level is more natural and clear. That is, we can directly measure the degree of change by analyzing how many concepts are inserted or eliminated after reduction. To achieve this, we need to give a formal definition to the intuitive examples of "inserted concepts" and "eliminated concepts". Hence, we are hereby to give our formal definitions of inserted and eliminated concepts, explain why these are considered "unexpected", and discuss the possibilities of controlling their quantities.

Our definitions for the inserted concepts and eliminated concepts are as follows.

Definition 11. *Suppose that $T = (G, M, Y)$ is a formal context and $T' = (G, M, Y')$ is the reduced context generated from T by an object reduction. Then, each concept $(A, B) \in \mathrm{L}(T')$ is defined to be an inserted concept if there is no (A', B') in $\mathrm{L}(T)$ such that either of the following two conditions holds:*

- *(A, B) is a specialization of (A', B'). That is, $A \subseteq A'$ and $B \subseteq B'$, or*
- *(A, B) is a generalization of (A', B'). That is, $A \supseteq A'$ and $B \supseteq B'$.*

Similarly, for each concept $(A, B) \in \mathrm{L}(T)$, it is defined to be an eliminated concept if there is no (A', B') in $\mathrm{L}(T')$ such that either of the above two holds.

This definition is naturally deduced from the basic ideology of FCA – in FCA, if we find an objects-attributes pair (G_1, M_1) completely included in another pair (G_2, M_2), *i.e.*, $G_1 \subseteq G_2$ and $M_1 \subseteq M_2$, we should consider the information contained in (G_1, M_1) compatible with that of (G_2, M_2), so we only extract the maximal (G_2, M_2) which cannot be included in another pair as the representative knowledge, *i.e.*, the formal concepts. Based on the same ideology, in concept lattice reduction, if we find a concept (G_3, M_3) in the reduced lattice is a generalization or specialization of a concept (G_4, M_4) in the original lattice, this should mean that the knowledge they represent is consistent, but only generalized or specialized. Instead, if (G_3, M_3) just overlaps with (G_4, M_4) but does not include or is included by (G_4, M_4), these should be considered representing different knowledge.

The properties of inserted concepts

We have found two properties of inserted concepts. First, it is clear that inserted concepts only occur when we apply the OR rule because when applying the AND rule, we will not introduce new relations into the context, so any concept in the reduced lattice will always be a specialization of some concept in the original lattice.

The second property is described in the following proposition.

Proposition 1. *Suppose an object reduction reduces the original context* $T = (G,M,Y)$ *into the reduced context* $T' = (G,M,Y')$ *applying the OR rule. Then, a concept* $(G_1,M_1) \in \mathrm{L}(T')$ *is an inserted concept only if there does not exist a concept* $(G_2,M_2) \in \mathrm{L}(T)$ *such that* $M_1 = M_2$.

Proof. Now assume $(G_1,M_1) \in \mathrm{L}(T')$ is an inserted concept and we have an $(G_2,M_1) \in \mathrm{L}(T)$. Since we only add new incidences but not remove any when we apply the OR rule, we should have $\mathrm{O}_{T'}(M_1) \supseteq \mathrm{O}_T(M_1) = G_2$. Since we also have $\mathrm{O}_{T'}(M_1) = G_1$, we should have $G_1 \supseteq G_2$, which implies that (G_1,M_1) is the generalization of (G_2,M_1) and contradicts the assumption. □

This property is important since it allows us to compute the number of inserted concepts. In the next section, we will show how to achieve this in our method.

The properties of eliminated concepts

We have also found two properties of eliminated concepts. First, it is clear that eliminated concepts only occur when we apply the AND rule because when applying the OR rule, we will not remove relations from the context, so any concept in the original lattice will always be generalized into another concept in the reduced lattice.

The second property is described in the following proposition.

Proposition 2. *Suppose an object reduction reduces the original context* $T = (G,M,Y)$ *to the reduced context* $T' = (G,M,Y')$ *applying the AND rule. Then, for any concept* $(G_1,M_1) \in \mathrm{L}(T')$*, there exists exactly one concept* $(G_2,M_2) \in \mathrm{L}(T)$ *such that* $G_1 \subseteq G_2$ *and* $M_1 \subseteq M_2$*. Moreever, we must have* $M_1 = M_2$.

Proof. Clearly, we must have a $(G_2,M_2) \in \mathrm{L}(T)$ such that $G_1 \subseteq G_2$ and $M_1 \subseteq M_2$ because when applying the AND rule, we only remove incidences. Now assume we have $M_1 \subset M_2$. Then, we must have an object subset $G_3 \subseteq G$ such that $G_3 \cap G_1 = \emptyset$ and $\mathrm{A}_{T'}(G_1 \cup G_3) = M_1$ since when we apply the AND rule, we always modify the attributes of a group of objects into their intersection. We should then have $\mathrm{O}_{T'}(M_1) \supseteq G_1 \cup G_3$, which contradicts the

assumption that $(G_1, M_1) \in \mathrm{L}(T')$ because it requires $\mathrm{O}_{T'}(M_1) = G_1$. Hence, we must have $M_1 = M_2$. □

Note that in the proposition, we only have $M_1 = M_2$ hold, but not $G_1 = G_2$. In the argument of the proof, if we assume $G_1 \subset G_2$, we cannot reach the dual relation $\mathrm{A}_{T'}(G_1) \supseteq M_1 \cup M_3$ which leads to contradicting the assumption. The reason why objects and attributes are not dual here is simply because the merging operation is not symmetric, that is, we only merge objects to make them have the same attributes, but do not merge attributes to make them be shared by the same objects.

This proposition suggests that when applying the AND rule, it is impossible for two concepts to be specialized into the same concept. In other words, the number of concepts reduced by an object reduction will equal the number of eliminated concepts. Hence, as long as we want to reduce the concept lattice, it is impossible to prevent the elimination of concepts completely. However, note that, unlike inserted concepts, eliminated concepts are not always unexpected. For example, among all three concepts eliminated in the example shown in Fig. 3.6, two of them – $(\{g_3\}, \{m_1, m_2, m_6\})$ and $(\{g_4\}, \{m_3, m_4, m_5, m_6\})$ may be considered as those representing the minor differences between objects, so it is proper to remove them. Hence, instead of completely preventing the occurrence of eliminated concepts, a better solution would be controlling the number of eliminated concepts, which makes for a trade-off between the effectiveness and the cost of the reduction. In the next section, we will show how to achieve this trade-off by using linear constraints to compute and control the number of eliminated concepts.

3.4 The Proposed ILP-based Method

In this section, we propose our new object reduction method. To explain our method precisely, we will use many mathematical expressions in this section. If readers find it difficult to understand, please refer to Section A.1 in the appendix, where we demonstrated two running examples to help better show the whole working flow of our method.

Our method uses the technique of integer linear programming (ILP). ILP is the technique that determines the optimal values of *integer variables* maximizing a target function under a group of *linear constraints* [19]. The canonical form of an ILP problem is shown below. Here $\mathbf{x}$, called the *variables*, is the vector to be decided, and $A, \mathbf{b}, \mathbf{c}$ are coefficient matrices or vectors that form the constraints:

$$\begin{aligned} &\textbf{maximize} && \mathbf{c}^T\mathbf{x} \\ &\textbf{subject to} && A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq 0 \textbf{ and } \mathbf{x} \in \mathbb{Z}^n. \end{aligned} \tag{3.1}$$

Besides numerical problems, ILP can also be used to solve logical problems, *i.e.*, to determine the optimal values of Boolean variables under a group of logical constraints. This is because if we represent TRUE with 1 and FALSE with 0, then a Boolean variable a would be equivalent to an integer variable satisfying that $0 \leq a \leq 1$, and all logical constraints can be translated into an equivalent group of numerical linear constraints. For example, assume that x_1 and x_2 are Boolean variables and B is a Boolean constant. The logical constraint $x_1 \wedge x_2 = B$ is equivalent to the following group of numerical linear constraints:

$$\begin{aligned} & x_1 + x_2 \leq 1 + B, \\ & x_1 \geq B, \\ & x_2 \geq B, \\ & 0 \leq x_1, x_2 \leq 1. \end{aligned} \tag{3.2}$$

It has been proven that all types of Boolean operators can be translated into numerical linear operators. For more details, please refer to [19]. Hence, in the latter part of the paper, all constraints are directly written in the logical form instead of their equivalent numerical form for convenience.

We chose ILP for solving the object reduction problem mainly because FCA has a strong connection with formal logic. That is, we can regard the formal context $T = (G, M, Y)$ as a Boolean matrix $\mathbf{X} = \{x_{ij}\}$, where $x_{ij} = \text{TRUE}$ if $(g_i, m_j) \in Y$ and $x_{ij} = \text{FALSE}$ if $(g_i, m_j) \notin Y$. In this case, most properties of formal concepts can be translated into logical rules. For example, the most basic closure property of a formal concept (G_o, M_o), that is, $\mathrm{O}_T(M_o) = G_o$ and $\mathrm{A}_T(G_o) = M_o$, can be translated into the following Boolean equations:

$$\begin{aligned} \bigwedge_{g_i \in G_o} \left(\bigwedge_{m_j \in M_o} x_{ij} \right) \wedge \bigwedge_{g_i \notin G_o} \left(\neg \bigwedge_{m_j \in M_o} x_{ij} \right) = \text{TRUE}, \\ \bigwedge_{m_j \in M_o} \left(\bigwedge_{g_i \in G_o} x_{ij} \right) \wedge \bigwedge_{m_j \notin M_o} \left(\neg \bigwedge_{g_i \in G_o} x_{ij} \right) = \text{TRUE}. \end{aligned} \tag{3.3}$$

Note that although there are other methods for working out the values of a group of Boolean equations, we still choose ILP because we consider the Boolean equations for each object reduction task too complicated to give analytical solutions. Now it should be clear that we can solve the task of object reduction with ILP by transforming it into a logical optimization problem – the problem of finding the optimized value of the Boolean matrix $\mathbf{X}$ such that as many objects as needed can be reduced under a group of logical constraints that prevent the original content of the context, as well as the concept lattice, from being significantly

changed. The advantage of using ILP is also clear – we can not only know how many incidences we will modify directly from the Boolean matrix $\mathbf{X}$, but also use the logical rules above to check whether a concept will be changed after reduction. Furthermore, we found that the number of inserted and eliminated concepts may also be derived from a group of logical rules, which will be explained later in this section. Previous methods using numerical approaches can never achieve this because it is hard to link the features of formal concepts directly with non-logical approaches like SVD or Fuzzy-k-means. In other words, by using ILP, we can get complete control of what changes may be made to the context as well as the lattice, which solves the two problems of previous methods and will produce reduction plans with better quality.

In the latter of this section, we will present the detailed steps for translating the problem of object reduction into the canonical form of an ILP problem so that we can solve it with an ILP solver.

3.4.1 Target function

The primary target of object reduction is to reduce as many objects as possible [17, 26, 59]. Such a target can be expressed with the global target function

$$R = \sum_{i=1}^{|G|-1} \mathbf{1}_{\bigvee_{j=i+1}^{|G|} I_{ij}}, \tag{3.4}$$

where I_{ij} is the variable indicating whether objects g_i and g_j are identical after reduction, $|G|$ is the size of the original object set G, and $\mathbf{1}_B$ is the indicative function of Boolean expression B which equals 1 when $B = \text{TRUE}$ and 0 when $B = \text{FALSE}$.

This function counts the number of objects g_i having at least one identical opponent g_j where $j > i$, which equals the number of objects reduced. Thus, the global target of the model is to maximize R.

3.4.2 Variables

Theoretically, we may merge any groups of objects in the context. In that case, we would need to assign $|Y|$ variables when applying the AND rule or $|G| \times |M| - |Y|$ variables when applying the OR rule in total. However, we consider it better only to allow modifying part of them because assigning too many variables will increase the computational complexity of the logical optimization problem and slow down the solving process. To achieve this, we introduce a hyper-parameter ε_r for controlling the number of variables to be assigned.

Suppose that $\mathbf{X} = \{x_{ij}\}$ and $\mathbf{C} = \{c_{ij}\}$ are two $|G| \times |M|$ Boolean matrices, where x_{ij} is the Boolean variable representing whether the incidence (g_i, m_j) is TRUE in the original context; and c_{ij} is the Boolean variable representing whether we should assign a variable to allow incidence (g_i, m_j) to be modified. Then, c_{ij} is determined to be TRUE if there exists another object $g_k \in G$ such that all the following conditions are satisfied:

$$\begin{aligned} &|(\mathrm{A}_T(g_i) - \mathrm{A}_T(g_k)) \cup (\mathrm{A}_T(g_k) - \mathrm{A}_T(g_i))| \leq \varepsilon_r, \\ &x_{ij} = \neg B, \\ &x_{kj} = B, \end{aligned} \tag{3.5}$$

where B is a Boolean constant set to TRUE if we apply the OR rule and FALSE if we apply the AND rule. In other words, if modifying a certain incidence x_{ij} cannot contribute to making g_i merge with another object g_k such that the "distance" of g_i and g_k in the original context is smaller than ε_r, we would consider it valueless to modify x_{ij} and thus should set x_{ij} to be unmodifiable.

Note that in some extreme cases, this strategy may still fail to control the number of variables. For example, in the formal context shown in Table 3.1, when we apply the OR rule, even if we set $\varepsilon_r = 2$, we will still find all incidences x_{ij} such that $x_{ij} =$ FALSE are modifiable, as if we have set $\varepsilon_r = \infty$. Despite this limitation, we still consider this strategy useful because as we will introduce later in Section 3.5.3, in both real contexts and randomly-generated contexts, setting ε_r to a small value will effectively decrease the number of variables. Furthermore, even if such extreme cases occurs and we have assigned many variables, we can still ensure the context and the concept lattice are not modified too much by introducing more constraints which will be introduced later in this section. Hence, in such an extreme case, although the execution time of our method may become longer, the quality of the output that generated by our method will not be affected.

Table 3.1 A sample context for explaining the procedure of setting variables.

	m_1	m_2	m_3	m_4	m_5
g_1	×	×			
g_2		×	×		
g_3		×		×	
g_4		×			×

For each incidence x_{ij} such that $c_{ij} =$ TRUE, we assign a variable v_{ij} representing that whether the incidence should be modified or not.

3.4.3 Basic constraints

In the target function, we have defined a group of variables I_{ij} as indicators for whether object g_i and g_j will become identical after reduction. The values of these variables for the task can be obtained with the XOR $\oplus$ and the XNOR $\odot$ operators, which are represented as a group of constraints:

$$t_{ij}^{(k)} = \begin{cases} x_{ik} \odot x_{jk} & \text{if } \neg c_{ik} \wedge \neg c_{jk} = \text{TRUE}, \\ (x_{ik} \odot x_{jk}) \oplus v_{ik} & \text{if } c_{ik} \wedge \neg c_{jk} = \text{TRUE}, \\ (x_{ik} \odot x_{jk}) \oplus v_{jk} & \text{if } c_{jk} \wedge \neg c_{ik} = \text{TRUE}, \\ (x_{ik} \odot x_{jk}) \odot (v_{ik} \odot v_{jk}) & \text{if } c_{ik} \wedge c_{jk} = \text{TRUE}, \end{cases} \tag{3.6}$$

$$I_{ij} = \bigwedge_{k=1}^{|M|} t_{ij}^{(k)}.$$

3.4.4 Constraints for preventing too much modification

We will get the total number of incidences modified by summing up the values for all v_{ij}. Hence, to prevent the context from being modified too much, we can add the following constraint:

$$\sum_{c_{ij}=\text{TRUE}} \mathbf{1}_{v_{ij}} \leq \varepsilon_m. \tag{3.7}$$

3.4.5 Constraints for controlling the number of inserted and eliminated concepts

As analyzed above, the eliminated concepts only occur when we apply the AND rule, and the inserted concepts only occur when we apply the OR rule. Hence, here we also discuss the two cases individually.

The AND rule

As suggested by Proposition 2, for every concept $(G_o, M_o) \in \mathrm{L}(T)$ from the original lattice, to check if it will be eliminated, we only need to check if M_o is still the intent of a concept in the reduced concept lattice, and this can be computed with a series of equations. First, for all $g_i \in G_o$, we are to compute S_{M_o,g_i}, which represents whether the object g_i will still have all attributes in M_o after reduction. Note that here we only need to compute the values of S_{M_o,g_i}

for objects g_i in G_o but not the whole object set G. This is because (G_o, M_o) is a concept in T, so for an object $g_i \notin G_o$, it cannot have all attributes in M_o even before reduction:

$$S_{M_o,g_i} = \bigwedge_{m_j \in M_o} \left(x_{ij} \wedge \neg \left(c_{ij} \wedge v_{ij}\right)\right). \tag{3.8}$$

Then, we are to compute $E_{(G_o,M_o)}$, representing whether none of the objects in G_o will have all attributes in M_o after reduction. This is one of the two cases in that M_o will not be an intent after reduction:

$$E_{(G_o,M_o)} = \bigwedge_{g_i \in G_o} \neg S_{M_o,g_i}. \tag{3.9}$$

Next, we are to compute $F_{(G_o,M_o)}$, representing whether the "featured" part that makes (G_o, M_o) a formal concept will be removed after reduction. That is, whether we will have $M_o \subset \mathrm{A}_T(\{g_i\})$ for all $g_i \in \mathrm{O}_{T'}(M_o)$. This is the other case in that M_o will not be an intent after reduction:

$$F_{(G_o,M_o)} = \bigvee_{m_j \notin M_o} \bigwedge_{g_i \in G_o} \neg S_{M_o,g_i} \vee \left(x_{ij} \wedge \neg \left(c_{ij} \wedge v_{ij}\right)\right). \tag{3.10}$$

Now we can compute $D_{(G_o,M_o)}$, representing whether concept (G_o, M_o) will be an eliminated concept:

$$D_{(G_o,M_o)} = E_{(G_o,M_o)} \vee F_{(G_o,M_o)}. \tag{3.11}$$

Finally, by summing up the value of $\mathbf{1}_{D_{(G_o,M_o)}}$, we will get the total number of eliminated concepts. As previously analyzed, not all eliminated concepts are considered unexpected. Hence, we can add the following constraint, which sets the maximum allowable number of eliminated concepts to ε_I to keep a certain degree of reduction rate and prevent the concept lattice from being changed too much:

$$\sum_{(G_o,M_o) \in \mathrm{L}(T)} \mathbf{1}_{D_{(G_o,M_o)}} \leq \varepsilon_I. \tag{3.12}$$

The OR rule

Inserted concepts are in the lattice of the reduced context, which is unknown before we generate the full reduction. Hence, theoretically, we need to enumerate all attribute sets $M_a \in \mathrm{P}(M)$ and check if each set will be the extent of an inserted concept with the closure property. Here $\mathrm{P}(M)$ is the power set of M. However, this is unnecessary because only a part of the incidences in the context are modifiable. For an attribute set M_1, if all modifiable

incidences are modified and there is still no object $g \in G$ that satisfies $\mathrm{A}_{T'}(g) \supseteq M_1$, then M_1 should never be an extent of a concept in $\mathrm{L}(T')$. Furthermore, as analyzed in Section 3.3.2, if an attribute set M_2 is an extent of a concept in $\mathrm{L}(T)$, the attribute set should also be impossible to be an extent of a concept in $\mathrm{L}(T')$. From the above, it is better for us to first determine the candidate space of attribute sets which are possibly an extents of concepts in $\mathrm{L}(T')$.

Let us denote the candidate space as $\mathrm{Can}(M)$. As analyzed above, to determine $\mathrm{Can}(M)$, we need to find all attribute sets that are supported by at least one object, which is equivalent to the task *frequent item set mining* and can be accomplished with algorithms like *Apriori* [10]. After determining $\mathrm{Can}(M)$, we can enumerate each $M_a \in \mathrm{Can}(M)$ and check whether the attribute set M_a will be the intent of an inserted concept after reduction.

Let U_{M_a} represent whether the attribute set M_a will be the intent of an inserted concept after reduction. We can also compute the value of U_{M_a} with a series of equations. First, we are to derive S_{M_a,g_i} for every object $g_i \in G$, which represents whether the object g_i will have all attributes in M_a after reduction as follows:

$$S_{M_a,g_i} = \bigwedge_{m_j \in M_a} \left(x_{ij} \vee \left(c_{ij} \wedge v_{ij}\right)\right). \tag{3.13}$$

Then, we are to compute E_{M_a}, representing whether M_a will become an intent of a concept after reduction. This is accomplished by checking the closure property of a formal concept. That is, to check whether $\mathrm{A}_{T'}(\mathrm{O}_{T'}(M_a)) = M_a$:

$$E_{M_a} = \bigwedge_{m_j \notin M_a} \left(\bigvee_{g_i \in G} S_{M_a,g_i} \wedge \left(\neg x_{ij} \wedge \neg \left(c_{ij} \wedge v_{ij}\right)\right) \right). \tag{3.14}$$

Next, we are to compute $F_{M_a,(G_o,M_o)}$, representing whether a concept (G_o, M_o) in the original concept lattice will be generalized to have the intent M_a after reduction. According to the definition of a generalized concept, this is accomplished by checking whether $G_o \subseteq \mathrm{O}_{T'}(M_a)$ and $M_o \subseteq M_a$ are both satisfied:

$$F_{M_a,(G_o,M_o)} = \begin{cases} \text{FALSE} & \text{if } M_o \not\subseteq M_a, \\ \bigwedge_{g_i \in G_o} S_{M_a,g_i} & \text{if } M_o \subseteq M_a. \end{cases} \tag{3.15}$$

Finally, according to the definition, M_a will become the intent of an inserted concept if it is not a generalization of any existing concepts. Hence, we can now compute U_{M_a} as follows:

$$U_{M_a} = E_{M_a} \vee \neg \left(\bigvee_{(G_o,M_o)\in \mathrm{L}(T)} F_{M_a,(G_o,M_o)} \right). \tag{3.16}$$

We can now add the following constraint to set the maximum allowed number of inserted concepts to ε_I:

$$\sum_{M_a \in \mathrm{Can}(M)} \mathbf{1}_{U_{M_a}} \leq \varepsilon_I, \tag{3.17}$$

Note that although, as analyzed in Section 3.3.2, the inserted concepts are always unexpected, sometimes we may still need to make a trade-off between the effectiveness of the reduction and the cost of the reduction. For example, if a reduction plan can reduce a large number of concepts at the cost of introducing only one inserted concept, it may still be considered an acceptable reduction plan. Hence, instead of completely preventing any introduced concepts from occurring, setting the maximum allowed number of inserted concepts would be a better option.

3.5 The experiments

We conducted two experiments to verify the theoretical proofs above that our methods can really control the number of modified incidences and inserted/eliminated concepts. The data used by the experiments are either randomly generated or taken from the "soybean" data provided by OpenML [106]. To the furthest of our knowledge, the numbers of objects and attributes of real contexts used in previous object reduction tasks usually range from 20 to 100 [17, 26, 59, 100]. Hence, in our experiments, the size of the smallest random context is set to 20×20 and that of the largest is set to 100×100. For the real soybean data, the nominal attributes are expanded into Boolean attributes[3] and the repeated objects are removed from the data[4]. After processing, the soybean data has 53 objects and 107 features with a density of 34%.

All scripts are written in Python. The previously proposed methods are implemented based on our own comprehension of the descriptions from the literature. Our method is implemented with CVXOPT [3]. The solvers we use for the ILP problem include CBC [33]

[3]For example, for an attribute A taking the values (a,b,c), we create 3 attributes Aa, Ab, Ac for representing whether the attribute A of an object is a, b or c, respectively.

[4]For that removing these objects will not change the concept lattice.

and Gurobi [41]. We run all the scripts on a Ubuntu 18.04 system with 93.00GB RAM and a 2.4GHz CPU.

3.5.1 Experiment 1: controlling the number of incidences modified

In this experiment, we will test whether our method can generate better reduction plans which reduces the same number of objects and concepts as previous methods but modifying fewer incidences. The test data for this experiment include the full soybean dataset and a randomly generated context with 50 attributes and 50 objects. Previous methods we study include the well-preferred SVD method [17] and the fuzzy k-means method [59]. For our method, the constraints for controlling inserted/eliminated concepts are not added in this experiment. We set different parameters for these methods and evaluate their performances using the following three metrics. To find the meanings of previous methods' parameters, please refer to the corresponding publications.

- The **object reduction rate**. That is, the percentage of reduced objects. Suppose we have n "unique" objects before reduction and n' objects after reduction, the object reduction rate would be $\frac{n-n'}{n}$.
- The **concept reduction rate**. That is, the percentage of reduced concepts. Suppose we have x concepts in the concept lattice before reduction and y concepts after reduction, the concept reduction rate would be $\frac{x-y}{x}$.
- The **context fidelity**. That is, the percentage of **unmodified** incidences. Suppose we modified k incidences in an $n \times m$ context after reduction, the context fidelity would be $1 - \frac{k}{nm}$.

The results are plotted in Fig. 3.7. From the figure, we find that clearly, our method can generate reduction plans with fewer modified incidences. Furthermore, previous methods' object and lattice reduction rates increase sharply and the context fidelity drops sharply when the similar rate threshold (or the *thres* value in the fuzzy k-means method) exceeds a tipping point. Those factors of our method, however, vary at a slower pace as we tune the parameter ε_r and a much slower pace as we adjust ε_m, showing that our model is more "controllable" than the previous methods. That is, compared to previous methods, we can fine-adjust the parameters in smaller steps to get the proper reduction rates we want to make a good trade-off between the effectiveness and the cost of the reduction plan.

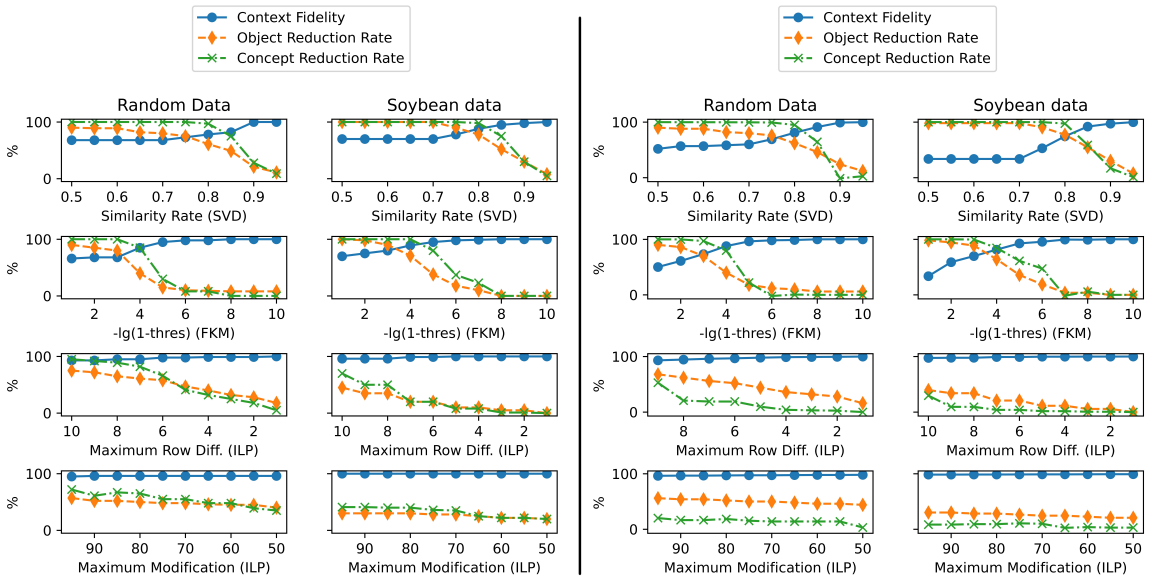


Fig. 3.7 Left: Statistics of the three methods on two datasets applying the OR rule. Right: Statistics of these methods on the same datasets applying the AND rule. For the SVD method, the rank of the reconstructed matrix is set to $3/4$. For the FKM method, the m value is set to 1.05. $\lg(\cdot)$ in the captions of the second row means $\log_{10}(\cdot)$. For our model, ε_r is set to 6 for the test cases shown in the fourth row, and the ε_m is set to infinite for the cases shown in the third row.

3.5.2 Experiment 2: controlling the number of eliminated/inserted concepts

In this experiment, we will test whether our method can control the number of eliminated/inserted concepts, with the special constraints added. Before the experiment, we first verify whether inserted/eliminated concepts will occur in a real object reduction task. In case we apply the AND rule, we have proven that all reduced concepts are eliminated concepts, so these will definitely occur. In case we apply the OR rule, the percentages of inserted concepts of the reduction plans generated by our method and the previous methods in Experiment 1 are plotted in Fig. 3.8. From the figure, it is clear that inserted concepts also commonly occur in real object reduction tasks.

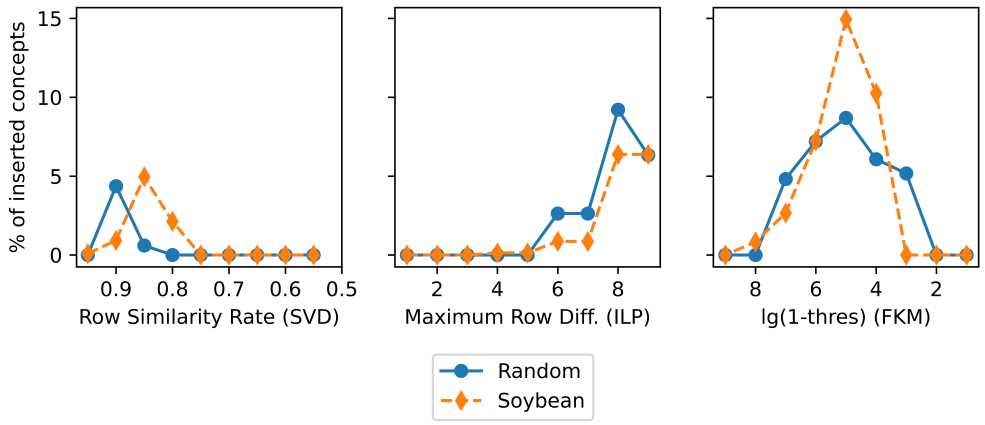


Fig. 3.8 The percentage of inserted concepts in the reduced concept lattice of different methods applying the OR rule in Experiment 1.

Controlling the number of inserted concepts

We generate four small 28×20 contexts and two large 100×100 contexts with a density of 15% as six test cases and run our method applying the OR rule. The small test cases are numbered from #1 to #4, and the large test cases are numbered #L1 and #L2. For all small test cases, ε_r is set to 6, and ε_m is set to infinite; for both large test cases, ε_r is set to 12, and ε_m is also set to infinite. The metrics we used in this experiment include the previously-introduced *object reduction rate* and *concept reduction rate* as well as *the number of inserted concepts* in the reduced concept lattice, which measures the reliability of the reduction in another aspect – the fewer number of inserted concepts we have, the more reliable the reduction is.

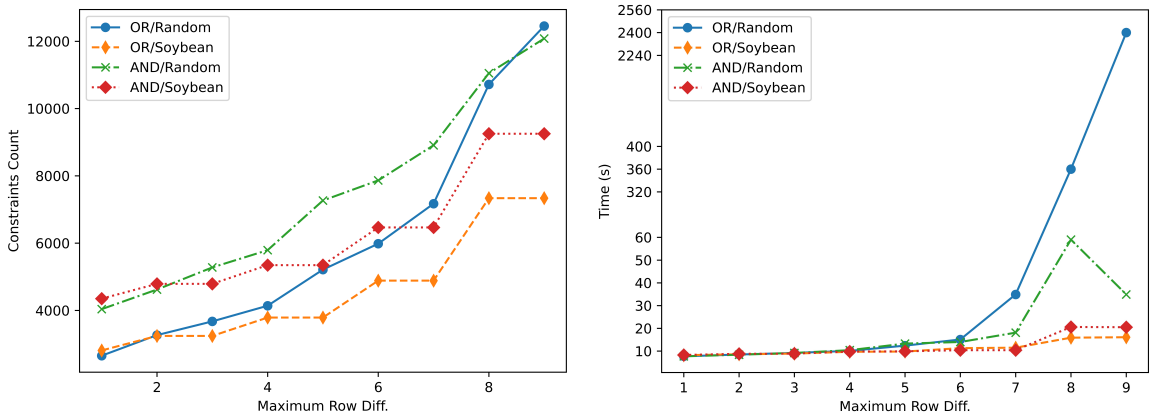


Fig. 3.9 The run time and the number of variables of our method in Experiment 1.

The experimental results for the four small contexts are plotted in Fig. 3.10 and those for the large contexts are plotted in Fig. 3.11.

From the figures, we find that our method can clearly control the number of inserted concepts to any level as expected. Furthermore, we find that generally, in the four small test cases, the reduction rates will increase as we tune ε_I to a higher value. However, in the two large data cases, when ε_I exceeds a specific value, the concept reduction rate begins to drop, eventually to a negative value. That is, the case that the number of concepts does not decrease but instead increases after reduction, which we discussed in Section 3.3, might actually occur in a practical object reduction task. Our analysis is that this occurs because as we start tuning the ε_I from a small value, the number of inserted concepts is still limited. In this case, a higher ε_I may give us more choices for merging objects, which will increase the reduction rate at first. However, as ε_I continues to grow, the number of inserted concepts increases sharply, causing the number of concepts in the reduced lattice to grow sharply as well. As a result, the "reduced" lattice may eventually contain more concepts than the original one. This shows the importance of controlling the number of inserted concepts. As analyzed in Section 3.3, to the furthest of our knowledge, none of the previous methods has the ability to control this number, which shows the advantage of our method.

Controlling the number of eliminated concepts

We generate four small 20×20 contexts and two large 100×100 contexts with a density of 15% as six test cases and run our method applying the AND rule. The small test cases are numbered from #5 to #8, and the large test cases are numbered #L3 and #L4. For all small test cases, ε_r is set to 3; for both large test cases, ε_r is set to 10. The metrics we used in this

experiment include the previously-introduced *object reduction rate* and *concept reduction rate* as well as *the number of eliminated concepts* in the original concept lattice, which measures the reliability of the reduction in another aspect – the fewer number of eliminated concepts we have, the more reliable the reduction is.

The results for the small contexts are plotted in Fig. 3.12 and those for the large contexts are plotted in Fig. 3.13. In the figures, the left column shows the statistics for our method without adding the constraints for controlling the number of eliminated concepts – we must control them only by tuning the parameter ε_m. The right column shows the statistics for our method with those constraints added, with ε_m set to infinite.

From the results, we can find that the number of eliminated concepts, *i.e.*, reduced concepts generally decreases as we tune ε_m to prevent less-similar object pairs from being merged. By adding the constraints for setting the maximum allowed number of eliminated concepts, we may find many more reduction plans at different levels of concept reduction rates. This is because our basic model, as well as all previous methods, can only output the *optimal* reduction, *i.e.*, the reduction plan that achieves the maximum object reduction rate. By adding the constraints for limiting the number of eliminated concepts, however, we can get alternate reduction plans having equal or lower object reduction rates but with the concept reduction rate at any level. We conclude that our method can better control the number of eliminated concepts to any level as expected.

3.5.3 Studies on the efficiency

We plot the number of variables and constraints, and the run time statistics of our method in the test cases of Experiment 1 in Fig. 3.9 and those of the test cases of Experiment 2 in Table 3.2. Note that in Table 3.2, "Size" means the size of the input context, and "Rule" means the merging rule for this test case. For each test case, the longest, shortest, and average run time among all different ε_I settings are reported. The run time statistics are all shown in seconds.

From these results, we find that generally, the run time of our method increases as the size of the context and the number of constraints grow. However, there are also some exceptions. For example, the number of constraints in test case #6 is smaller than that of test case #5, but the average run time of the former test case is longer than the latter one. Also, in some test cases like case #2, the run time varies greatly with different ε_I settings. Our analysis is that besides the number of variables and constraints, some other factors may also affect the difficulty of a linear optimization problem, such as the distances from the initial values of the variables to their optimal values [19]. Nevertheless, the size of the input context, which is directly tied to the number of constraints, is still the dominant factor, as the run times of

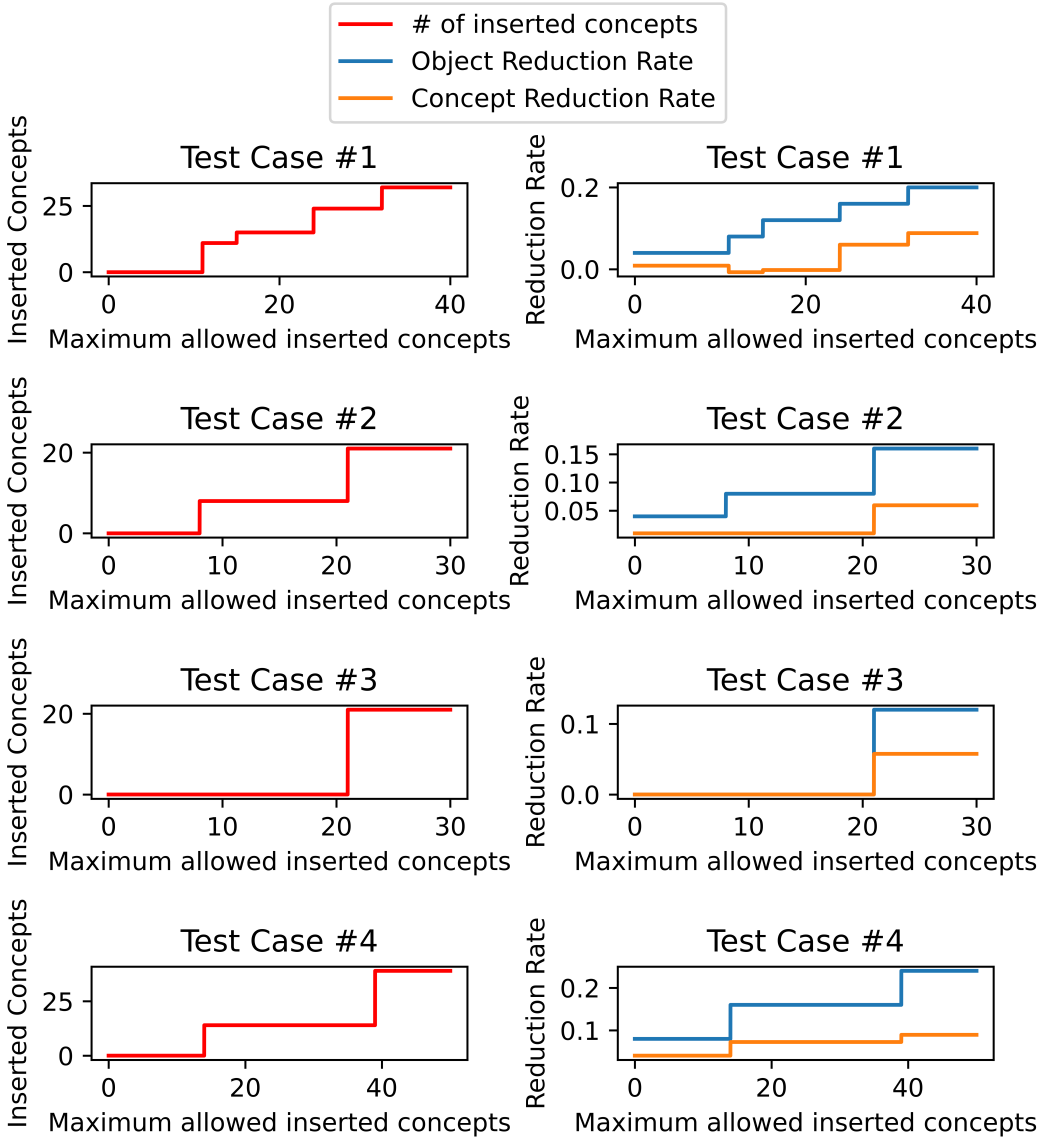


Fig. 3.10 The reduction rates and number of inserted concepts of our method on the four small test cases applying the OR rule, with different settings for the maximum allowed inserted concepts.

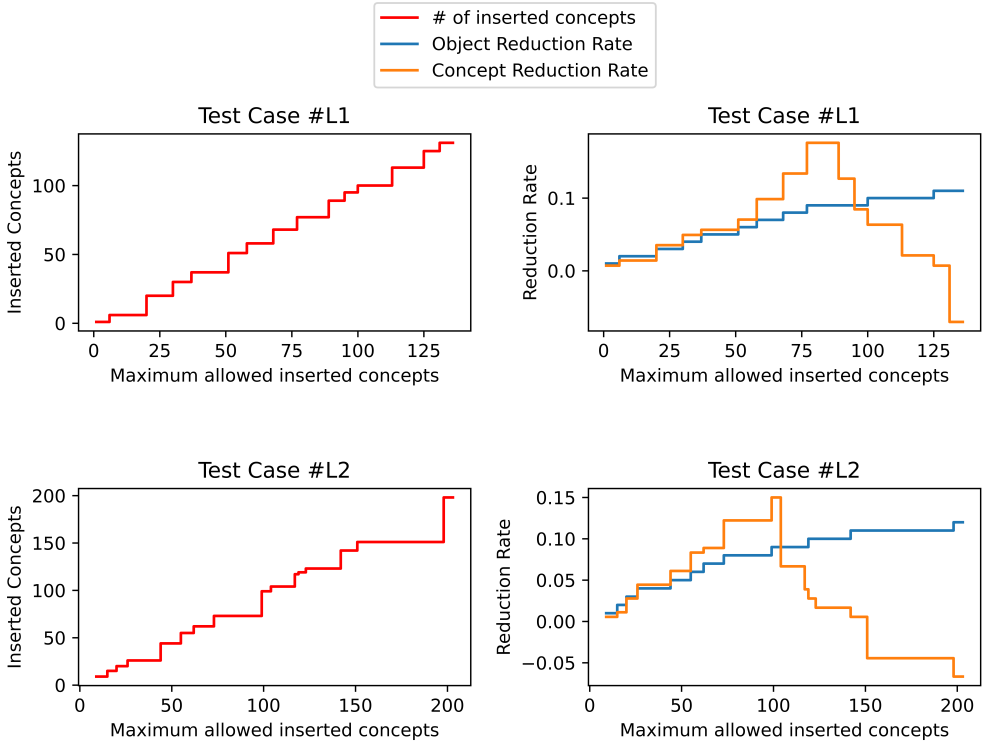


Fig. 3.11 The reduction rates and number of inserted concepts of our method on the two large test cases applying the OR rule, with different settings for the maximum allowed inserted concepts.

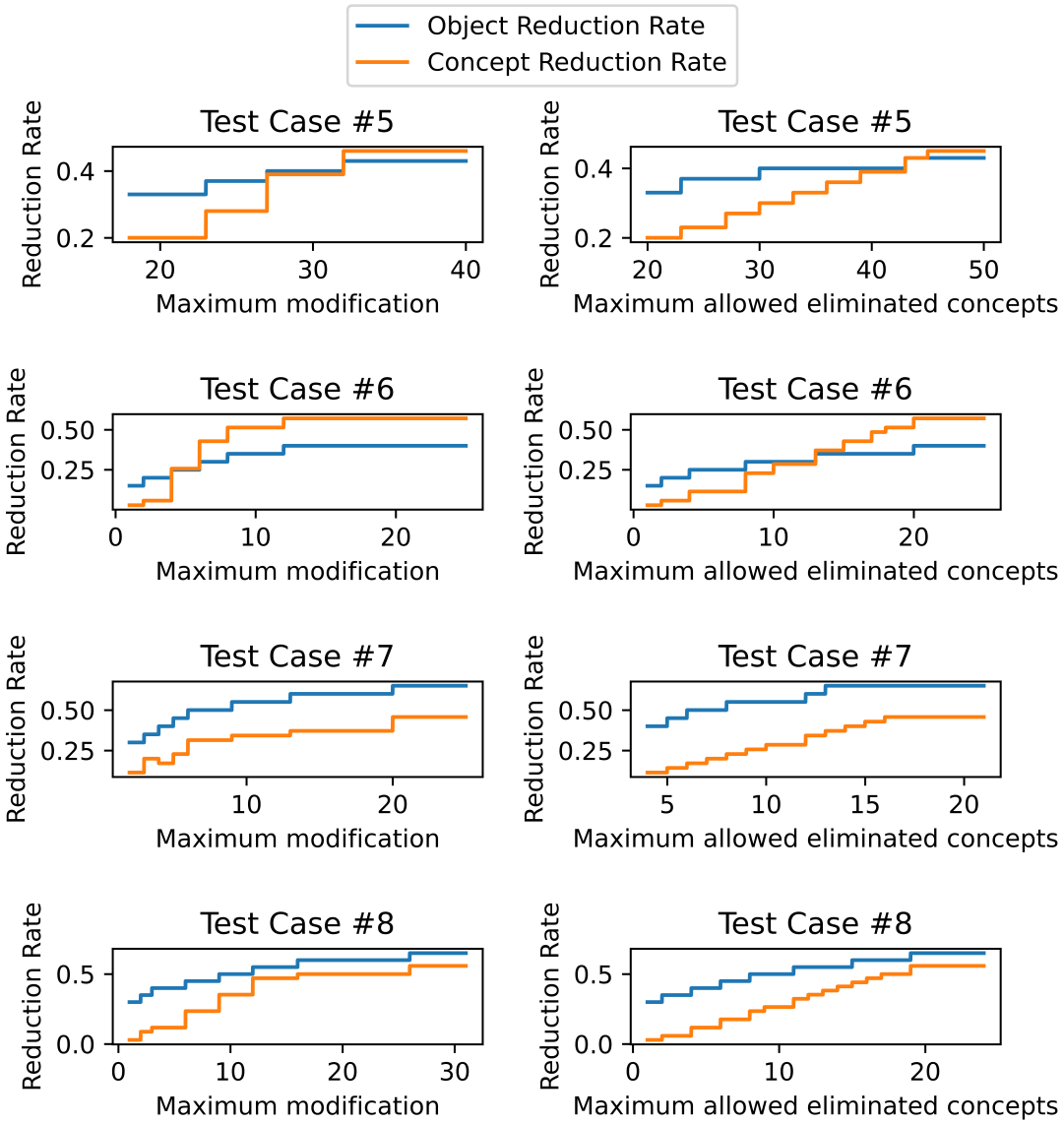


Fig. 3.12 The reduction rates of our method on the four small test cases applying the AND rule, with different settings for the maximum allowable modifications and maximum allowed eliminated concepts. Note that the concept reduction rate equals the percentage of eliminated concepts.

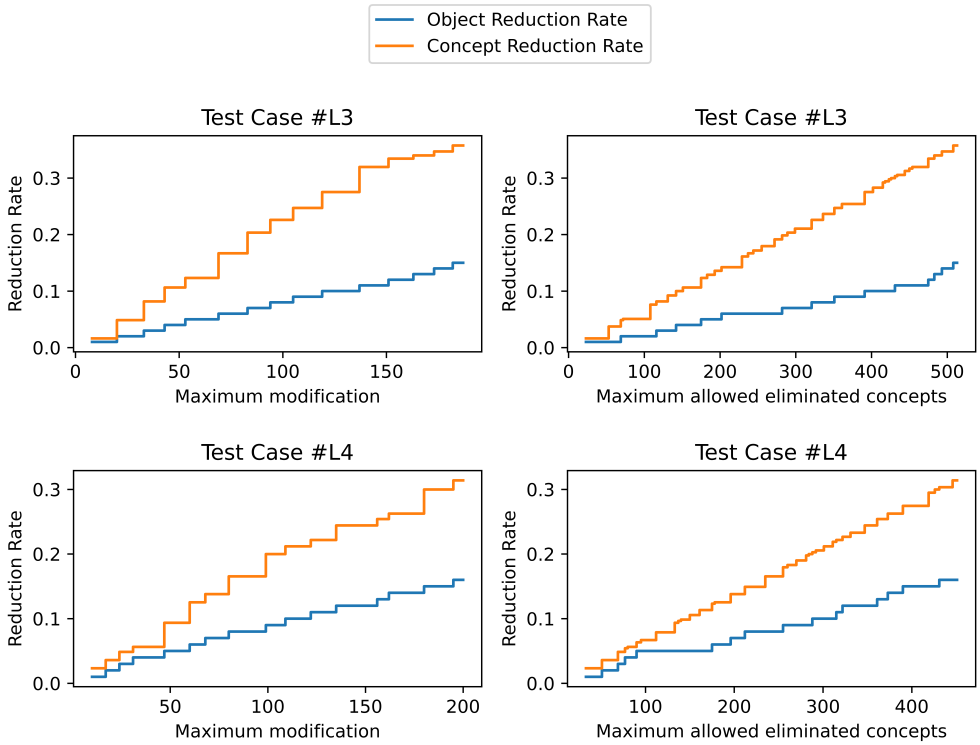


Fig. 3.13 The reduction rates of our method on the two large test cases applying the AND rule, with different settings for the maximum allowable modifications and maximum allowed eliminated concepts. Note that the concept reduction rate equals the percentage of eliminated concepts.

Table 3.2 The run time of our method in Experiment 2.

Case	Rule	Size	Constraints	Run time (s)		
				MAX	MIN	AVG
#1	OR	28×20	100,232	571	272	301.2
#2	OR	28×20	130,792	509	4	173.6
#3	OR	28×20	139,487	605	235	420.0
#4	OR	28×20	142,188	123	2	51.6
#L1	OR	100×100	2,098,356	7561	4548	6233.7
#L2	OR	100×100	2,810,899	7882	5253	6683.1
#5	AND	20×20	19,292	45	30	38.3
#6	AND	20×20	17,716	622	232	370.0
#7	AND	20×20	14,865	113	16	66.6
#8	AND	20×20	12,698	5	4	4.6
#L3	AND	100×100	983,530	5280	4476	5080.1
#L4	AND	100×100	1,023,350	5650	5056	5140.3

all four large test cases are much longer than those of the eight small cases. Generally, the maximum size that our method can finish running in a reasonable time is estimated to be around 100×100, because the running time of our method on contexts with such a size in Experiment 2 already has several hours. However, we consider it worth the extra time to prevent the context and the concepts from being excessively modified because otherwise, we may generate low-quality reduction plans that have no practical value at all.

3.6 Conclusion of the chapter

We propose a new object reduction method using integer linear programming. Our method can control the number of modified incidences and inserted/eliminated concepts which is not accomplishable by previous methods. These advantages enable our method to generate reduction plans that comply more closely to the basic principle of concept lattice reduction or namely that the reduction process should not cause significant changes in the original context and the knowledge in the concept lattice. For FCA4SML methods, such advantages of our method are of vital important because the results of the post-processing SML method relies heavily to the quality of the input. Hence, we believe our method can fit well into an FCA4SML method.

To further exploit this topic, we plan to combine our method with the technique of *iceberg concept lattice* [103] in order that our method can work with larger contexts. We also plan to study the cases when applying more complicated merging rules other than the global AND rule and OR rule. Finally, as our long-term goal, we plan to study the possibility of finding

an alternate method other than ILP that can control the number of modified incidences and the inserted/eliminated concepts during generation of reduction plans.

Chapter 4

An FCA4SML method which avoids the direct processing of formal concepts

4.1 Introduction

In this chapter, we propose an FCA4SML method which can achieve a good performance without directly processing the extracted formal concepts with an SML method. This is our second solution for the second issue that makes it hard to work out an FCA4SML method, *i.e.*, the number of formal concepts is so large that the concepts are difficult to be further processed by an SML method.

In the previous chapter, we have shown the possibility of using our ILP-based method to reduce the number of objects and formal concepts in the concept lattice. The method is useful in case the post-processing SML method directly takes the entire concept lattice as input. However, we may also consider utilizing an extra process first to generate an intermediate set from the formal concepts. Then, the intermediate set is processed by an SML method to get the final output. Since the size of the intermediate set is adjustable, by applying such a strategy, we may also prevent the input size of the SML process from being excessively large, and thus solve the issue. The content of the intermediate set is task-specified – the elements of the set do not need to be formal concepts and form into a concept lattice, but only need to fit the input format of the SML process, which means that this solution can save much time cost in maintaining the information within the concept lattice (*e.g.*, to control the number of inserted concepts and eliminated concepts). Accordingly, this solution is not generally applicable like the solution in the previous chapter as we have to develop a new method to generate the intermediate set for every different task. Hence, both solutions have their merits and demerits, and we should choose and apply either of them based on our actual needs.

Since the generation for the intermediate set has to be task-specified, in this chapter, we focus on the task *bipartite link prediction*. *Bipartite link prediction* is the task of predicting the absence or presence of unobserved links in a *bipartite network* [38, 70, 115]. A bipartite network is a structure consisting of two disjoint sets of nodes and a set of edges where every edge only connects two nodes from different sets. Many real-world relational data can be naturally modeled as bipartite networks where the two sets of nodes represent two groups of *entities* and the edges represent their *links* or *relations* [4, 97]. In real-world bipartite networks, some links may be missing or have not been observed yet [38, 115]. This leads to the need to predict whether an unobserved link should be a potential new one and gives rise to the task of *bipartite link prediction*. Making such a prediction based on the observed data of the network is proven possible thanks to the conclusion that any two nodes connected with a link should be similar to each other in terms of the connectivity features[62, 71, 109]. For example, given a protein-protein interaction network, we can predict potential unrecorded new interactions based on the topological structure of the network because two proteins interact with the same group of proteins, they should also have a high possibility of interacting with each other[5]. Based on this idea, various methods have been developed, and the task of predicting unobserved links is named the link prediction problem [71, 109]. Recently, the link prediction problem has become an extensively researched topic due to its high practical value. It has found various applications to different scenarios like recommending potential friends from a social network service [109], completing a knowledge graph [87], discovering possibly related research papers from a research network [13], or forecasting possible interactions between chemicals, genes, and diseases [32, 73].

As a typical machine learning task, the methods for bipartite link prediction can be categorized into two groups – the RML methods[51, 60, 121] and the SML methods. The RML methods for bipartite link prediction usually apply a similarity-based scoring strategy [120]. These first give a similarity score for all node pairs based on the information of the observed part of the network. Then, a new link is predicted between each node pair where their similarity score exceeds a certain threshold[62, 121]. Some similarity measures are based on only the local features of the network, such as *common neighbors* (CN) [110], the *Jaccard coefficient* (JC) [56, 121], the *Adamic-Adar coefficient* (AA) [115, 121], and the *preferential attachment* (AA) [7, 121]; others are based on the global features of the whole network, such as *random walk with restart* (RWR) [6] and *PageRank* [83].

The SML methods, on the other hand, treat the observed part of the network as training samples and use algorithmic approaches to train a model, and use the model for predicting unobserved links [15, 54, 96, 99, 120]. One of the most influential SML bipartite link prediction methods is *matrix factorization* (MF) [88, 110, 114, 123], which is also called

collaborative filtering The basic idea of MF is to extract latent factors from the bi-adjacent matrix of the observed part of the network and try to reconstruct the whole network with these latent factors. Here, the bi-adjacent matrix A of a bipartite network $G = (V_1, V_2, E)$ is a $|V_1| \times |V_2|$ matrix, where a_{ij}, the element on the i-th row and j-th column of the matrix represents the link status between node v_1 and v_j. Given such an $N \times M$ matrix A, an MF-based link prediction method aims at decomposing it into the product of a $N \times k$ matrix P and a $k \times M$ matrix Q, where k is far smaller than M. Since the ranks of both P and Q are far smaller than that of A, the product of P and Q is considered to be a low-dimensional approximation of the original matrix, which hopefully will abstract the unnecessarily detailed information and keep the essential information of the original matrix A. Hence, for each node pair (v_i, v_j) where the link is unobserved, the corresponding value of $(PQ)_{ij}$ is considered to be the confidence score or possibility of whether there should be a potential link. That is, a link is predicted between the node pair if the score exceeds a certain threshold. See Fig 4.1 for a clearer depiction of the working flow of an MF-based bipartite link prediction.

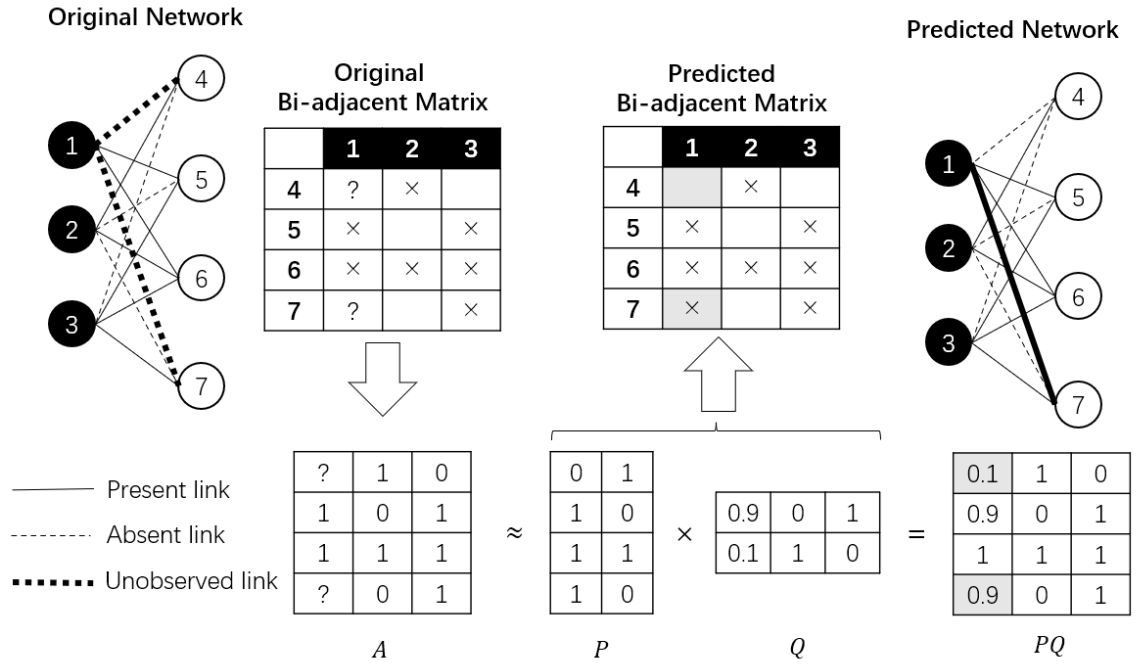


Fig. 4.1 An example of the working flow of the MF-based bipartite link prediction. In the bi-adjacent matrices, present links, absent links, and unobserved links are represented with crosses, empty cells, and question marks, correspondingly. The bi-adjacent matrix of the original network is converted into a numerical matrix A, which is then decomposed into the product of P and Q. In the reconstructed matrix PQ, the confidence values of two node pairs with unobserved links (v_1, v_4) and (v_1, v_7) are 0.1 and 0.9 separately, so finally, we only predict a new link between (v_1, v_7).

Recently with the rapid growth of computing power, SML methods have taken up the majority of bipartite link prediction methods [62, 71, 88, 109]. However, compared to RML methods, they rely heavily on the ground truth of the link status in the observed part of the network to create supervised data [88]. In real-world bipartite networks, there is often no ground truth of an absent link. For example, in a chemical-disease interaction network, a chemical not linked to a disease does not mean that they do not interact with each other but only implies that their interaction is not yet observed. If a large number of these non-concrete absent links are treated as reliable negative training samples, it is highly likely to result in a "modest" model which tends not to predict new links at all. Such a problem has already been frequently spotted in previous research and was proved to influence prediction accuracy significantly [16, 81]. Till now, there have been three different strategies for solving this issue. The first strategy is to utilize some side information from other data sources to counterbalance the unreliable prediction score. The *NMF-LP method* applies this strategy [13]. The second strategy is to make a minor perturbation to the data, which is applied in [16]. The third strategy is to use an RML method first to make a preliminary prediction. Then, the prediction scores of the RML methods are used to initialize training samples used in the SML method or regularize the weights of the SML model. This solution is most widely applied because it does not require external information and is easy to implement SRNMF,matrix,mcimpute. Typical methods applying this strategy include *SRNMF* [110], which uses the *common neighbors* prediction scores to regularize the matrix factorization process and the methods introduced in [73], which apply various RML methods to initialize the matrix to be factorized.

In this chapter, based on the idea of the third strategy, we propose a new FCA4SML method utilizing a special pre-processing step called *negative sample selection* (NSS). The technique first uses a preliminary link prediction method utilizing FCA to mark out node pairs that are **least** likely to be potential links. Then, it randomly selects a certain percentage of negative samples from the unmarked node pairs. For the preliminary link prediction method, instead of applying a traditional node-similarity-based scoring strategy, our method first utilizes the *structure hole theory* [115] which gives preliminary predictions by extracting and analyzing the *overlapping bi-cliques* from the network. A similar method was proposed and studied in [121], while our method makes use of the theories and conclusions from the related field of FCA to reduce the time complexity from $O(|C|^2)$ to $O(|C|)$ where $|C|$ represents the number of all bi-cliques and that of overlapping bi-cliques, separately. Compared to those traditional scoring methods, which give scores based on local statistic features for every single node pair, our method focuses more on extracting and comparing the overall connectivity features of node clusters, which suggests that our method is robust against the bias caused by local features and is expected to have better performance. After this

well-designed negative sample selection procedure, we can get a more reliable intermediate set as the input of the preceding MF process and finally get our highly accurate prediction results. We conduct experiments on three real-world datasets to simulate two hypothetical application scenarios and found that our FCA4SML method can not only work but also outperform the raw MF method as well as all other previous unsupervised bipartite link prediction methods.

4.2 Preliminaries

4.2.1 Problem formulation and evaluation

This research studies the problem of bipartite link prediction on an input network $G_i = (V_{i1}, V_{i2}, E_i)$ and a target network $G_t = (V_{t1}, V_{t2}, E_t = E_t^+ \cup E_t^-)$, satisfying that $V_{i1} = V_{t1}, V_{i2} = V_{t2}$ and $E_i \subset E_t^+$ where E_t^+ and E_t^- represent the present links and absent links, separately. That is, the input network has the same nodes as the target network, while it only contains part of the present links and no ground truth information about absent links. The goal of link prediction is to use the information of G_i to build a predict network G_p. The more G_p appears similar to G_t, the more successful the system is.

To give a concise numerical evaluation, we use the following two measures: *AUC score* and *AUPR score*, which were applied in most previous research [13, 73, 114, 121, 123]. Both scores are estimated with the four basic values: TP, TN, FP, and FN. Here TP represents the number of samples that are actually positive (present) and is predicted positive; FP represents the number of samples that are negative (absent) but are falsely predicted to be positive; TN represents the number of samples that are actually negative and predicted negative; FN represents the number of samples that are actually positive but falsely predicted negative.

The AUC (Area Under the Curve) score is estimated by computing the area under the ROC (Receiver Operating Characteristic) curve. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. Here TPR and FPR are estimated as follows:

$$TPR \stackrel{\text{def}}{=} \frac{TP}{TP+FN},$$
$$FPR \stackrel{\text{def}}{=} \frac{FP}{TN+FP}.$$

The AUPR (Area Under the Precision-Recall curve) score is estimated by computing the area under the Precision-Recall curve, which is created by plotting the precision rate against the *recall* rate at various threshold settings. Here precision and recall are estimated as

follows:

$$Precision \stackrel{\text{def}}{=} \frac{TP}{TP+FP},$$
$$Recall \stackrel{\text{def}}{=} \frac{TP}{TP+FN}.$$

4.3 The proposed method

4.3.1 An overview of the working flow of our method

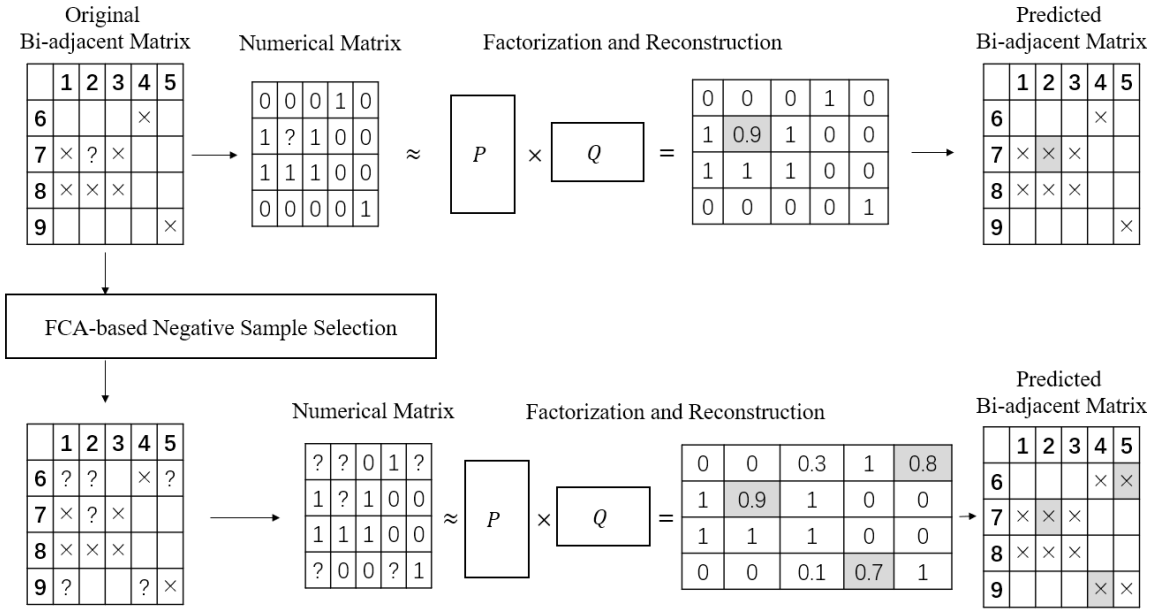


Fig. 4.2 The comparison of the working flow of the raw MF method and our joint method. The upper row depicts the working flow of the raw MF method, while the lower row depicts that of our method.

Our method is named *Matrix Factorization with Negative Sample Selection* (MF-NSS). It contains two parts – the negative sample selection and the MF-based link prediction. In the first part, it processes the network and extracts maximal bi-cliques using *formal concept analysis* (FCA), a technique originally proposed for ontology extraction but also strongly connected with network theory. Then, it picks out the node pairs which are least likely to be linked together. Next, it passes the bi-adjacent matrix of the input network, with all present links marked as positive samples and the aforementioned node pairs as negative examples, to the second part. In the second part, the matrix is factorized and approximated with a collaborative filtering process, and the node pairs where their corresponding score in the

reconstructed approximated matrix exceeds a threshold are outputted as our final predicted links. The comparison of the working flow of our method and the raw MF method is depicted in Fig 4.2.

4.3.2 Overlapping maximal bi-cliques and structural hole

In this research, we use a preliminary link prediction method utilizing the features of *bi-cliques*. A bi-clique $C = \{V_{c1}, V_{c2}, E_c\}$ of a bipartite network $G = (V_1, V_2, E)$ is a sub-network where there is a link between every node pair from different parts, that is, $V_{c1} \subseteq V_1, V_{c2} \subseteq V_2, E_c \subseteq E$ and for all $v_1 \in V_{c1}, v_2 \in V_{c2}$ we have $(v_1, v_2) \in E_c$. For such a bi-clique C, if no other bi-clique D is a super-network of C, then C is called a *maximal bi-clique*. See Fig 4.3 for an example of bi-cliques and maximal bi-cliques.

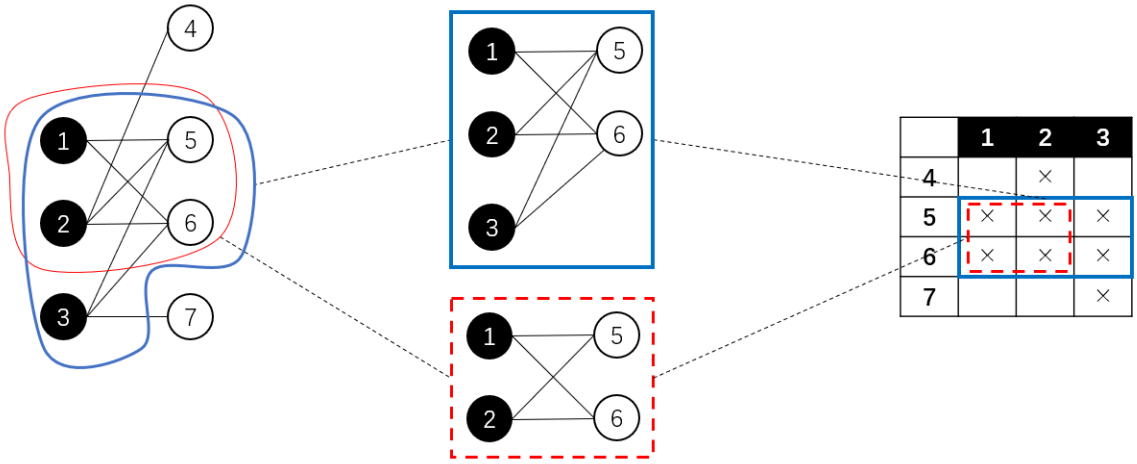


Fig. 4.3 An example of a bi-clique and a maximal bi-clique. The sub-network framed out in red from the bipartite network on the left is a bi-clique, which corresponds to the rectangle framed with red dash lines in the bi-adjacent matrix shown on the right. However, it is not a maximal bi-clique because it is a sub-network of another bi-clique framed in blue. The latter bi-clique is a maximal bi-clique, for it corresponds to the rectangle framed with solid blue lines on the right, which is a maximal rectangle box filled with crosses with rows and columns permutable.

Bi-cliques are considered clusters of strongly-related entities. For example, in an authorship network, a bi-clique may represent a group of co-researchers and their research; in a chemical-disease network, a bi-clique represents a group of similar chemicals and their affected diseases. Intuitively, if two clusters have a lot of nodes in common, it may imply that they are both parts of a larger cluster; that is, the unlinked node pairs from two clusters should have a high possibility of being linked together. This is the famous *structural*

hole theory [115, 121], which can be formalized into the following strategy for preliminary link prediction. Given a bipartite network G, two maximal bi-cliques $A = \{V_{A1}, V_{A2}, E_A\}$ and $B = \{V_{B1}, V_{B2}, E_B\}$ are overlapping bi-cliques if they satisfy $V_{A2} \subset V_{B2}$ which mutually implies $V_{B1} \subset V_{A1}$. To avoid biases, we add two extra coefficients – the component sizes of a bi-clique A, denoted as $s_1(A)$ and $s_2(A)$, separately; the overlapping rates of two bi-cliques A and B, denoted $\sigma_1(A,B)$ and $\sigma_2(A,B)$, separately, for measuring whether the overlapping bi-cliques are too trivial to represent a cluster of the nodes in a network:

$$
\begin{aligned}
s_1(A) &\stackrel{\text{def}}{=} |V_{A1}|, \\
s_2(A) &\stackrel{\text{def}}{=} |V_{A2}|, \\
\sigma_1(A,B) &\stackrel{\text{def}}{=} \frac{\min\{|V_{B1}|, |V_{A1}|\}}{\max\{|V_{B1}|, |V_{A1}|\}}, \\
\sigma_2(A,B) &\stackrel{\text{def}}{=} \frac{\min\{|V_{B2}|, |V_{A2}|\}}{\max\{|V_{B2}|, |V_{A2}|\}}.
\end{aligned}
$$

An overlapping maximal bi-clique pair A and B is considered non-trivial if the two bi-cliques have enough sizes and overlapping rates. That is, they should satisfy the following conditions:

$$
\begin{aligned}
\min\{s_1(A), s_1(B), s_2(A), s_2(B)\} &> \alpha. \\
\min\{\sigma_1(A,B), \sigma_2(A,B)\} &> \rho.
\end{aligned}
$$

where α and ρ are threshold parameters. Based on the structure hole theory, for a non-trivial overlapping maximal bi-clique pair, all node pairs in their structure hole, denoted as $H(A,B)$ are expected to have possible new links. That is, for all $v_1 \in V_{A1} - V_{B1}$ and $v_2 \in V_{B2} - V_{A2}$, a new link is predicted over each (v_1, v_2). See Fig 4.4 for a clearer view of this overlapping-based preliminary link prediction method.

Certainly, such an overlapping-based method can only weigh out some node pairs from others, which is not enough for a high-accurate link prediction method. However, it is already enough for our process of negative sample selection since we only need those node pairs that are **least** likely to be linked. That is, we simply apply this preliminary link prediction method to find out the node pairs that are **not** considered as possible links. These node pairs are marked as negative samples for the next part of our method, the MF-based link prediction, which will then extract the latent factors with these training samples and give a more accurate prediction.

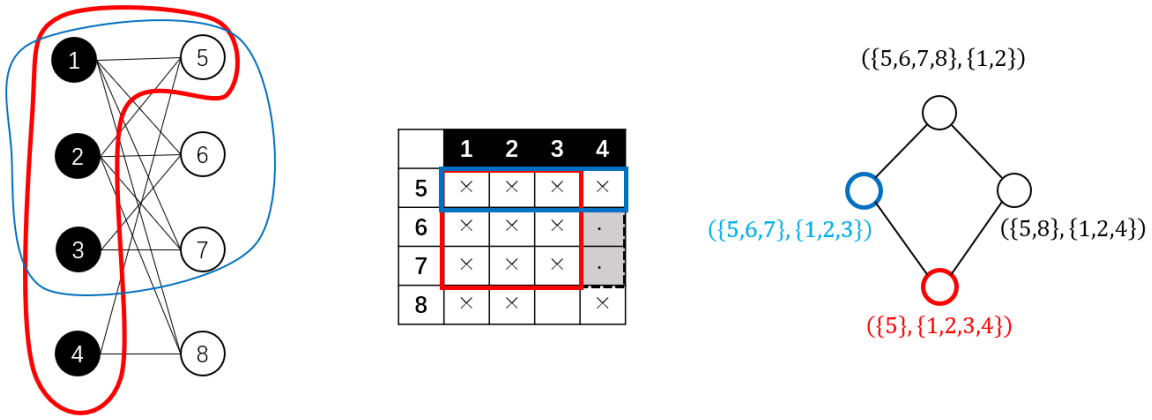


Fig. 4.4 An sample pair of overlapping bi-cliques and their structure hole. Left: A bipartite network with two overlapping maximal bi-cliques framed out in red and blue, separately. Middle: The bi-adjacent matrix of the network on the left. The aforementioned maximal bi-cliques correspond to the red and blue rectangles, separately. The gray cells filled with dots represent their structure hole. Right: the concept lattice representing all maximal bi-cliques from the network. The aforementioned bi-cliques correspond to the concepts marked out in blue and red, separately.

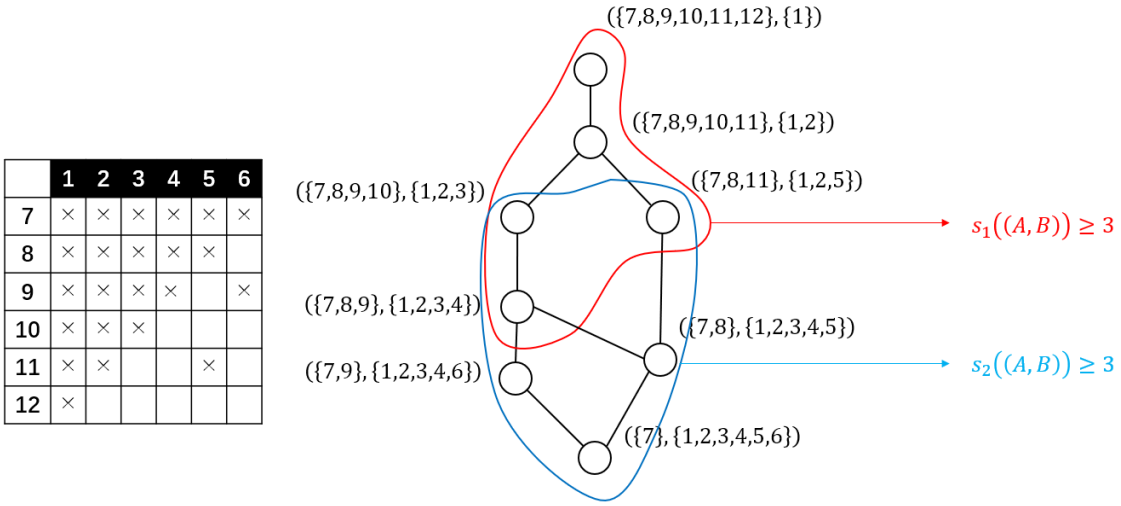
4.3.3 Maximal bi-cliques and FCA

Clearly that if we consider the two parts of a bipartite network as the object set and the attribute set, the bi-adjacent matrix will become a formal context, and a formal concept extracted from such a formal context should represent two maximal subsets of nodes from each part of the original network where every node pairs from two different part are linked together, which completely matches the definition of a maximal bi-clique. Furthermore, a pair of overlapping maximal bi-cliques can also correspond to two concepts (A_1, B_1) and (A_2, B_2) satisfying $(A_1, B_2) < (A_2, B_2)$, *i.e.*, a pair of concepts on the same path from the minimum concept to the maximum concept of the concept lattice, as is shown on the right of Fig 4.4. Furthermore, for three concepts (A_1, B_1), (A_2, B_2), and (A_3, B_3) satisfying $(A_1, B_1) < (A_2, B_2) < (A_3, B_3)$, we have the following deductions:

- $|A_1| > |A_2| > |A_3|, |B_1| < |B_2| < |B_3|$,
- $\sigma_1((A_1, B_1), (A_3, B_3)) = \sigma_1((A_1, B_1), (A_2, B_2)) \cdot \sigma_1((A_2, B_2), (A_3, B_3))$,
- $\sigma_2((A_3, B_3), (A_1, B_1)) = \sigma_2((A_3, B_3), (A_2, B_2)) \cdot \sigma_2((A_2, B_2), (A_1, B_1))$,
- $H((A_1, B_1), (A_3, B_3)) = H((A_1, B_1), (A_2, B_2)) \cup H((A_2, B_2), (A_3, B_3))$.

These deductions show that the component sizes and the overlapping rate of a pair of bi-cliques also correspond to the hierarchical features of the concept lattice. That is, all

bi-cliques with component sizes beyond the threshold correspond to the intersection part of an upward and a downward iceberg-shaped section of the lattice. All over-lapping bi-cliques with an overlapping rate beyond the threshold correspond to several chain-shaped sections of the lattice, where the structure wholes of each pair of concepts from the section are all included in that of the minimum and maximum concept of the section. See Figs 4.5 and 4.6 for a clearer depiction of such connections. All these connections above show that to extract all non-trivial overlapping bi-cliques, all we need is to enumerate all formal concepts and construct a concept lattice from the formal context corresponding to the bi-adjacent matrix.



	1	2	3	4	5	6
7	×	×	×	×	×	×
8	×	×	×	×	×	
9	×	×	×	×		×
10	×	×	×			
11	×	×			×	
12	×					

Fig. 4.5 An example of the iceberg-shaped sections of the concept lattice corresponding to the bi-cliques with component sizes exceeds a threshold 3.

Extracting overlapping bi-cliques with FCA algorithms

We extract maximal bi-cliques with a modified LCM algorithm [105]. LCM algorithm is an improved version of the prototype algorithm for FCA introduced in Section 2.2. Similar to the prototype algorithm, the LCM algorithm also uses the *augmentation operator* $\mathrm{Aug}((A_1,B_1),b)$ which generates a concept directly from a given concept (A_1,B_1) and an augmentation attribute m_b:

$$\mathrm{AUG}((A_1,B_1),b) \stackrel{\mathrm{def}}{=} ((B_1 \cup \{m_b\})^{(1)}, (B_1 \cup \{m_b\})^{(1)(2)})$$

Especially note that in the above equation, b is an *integer* ranges from 1 to $|M|$ representing the serial number of an attribute. As proven in Section 2.2, for any concept (A_1,B_1), all (A_2,B_2) satisfying that $(A_2,B_2) > (A_1,B_1)$ can generate with $\mathrm{AUG}((A_1,B_1),b)$ by choosing

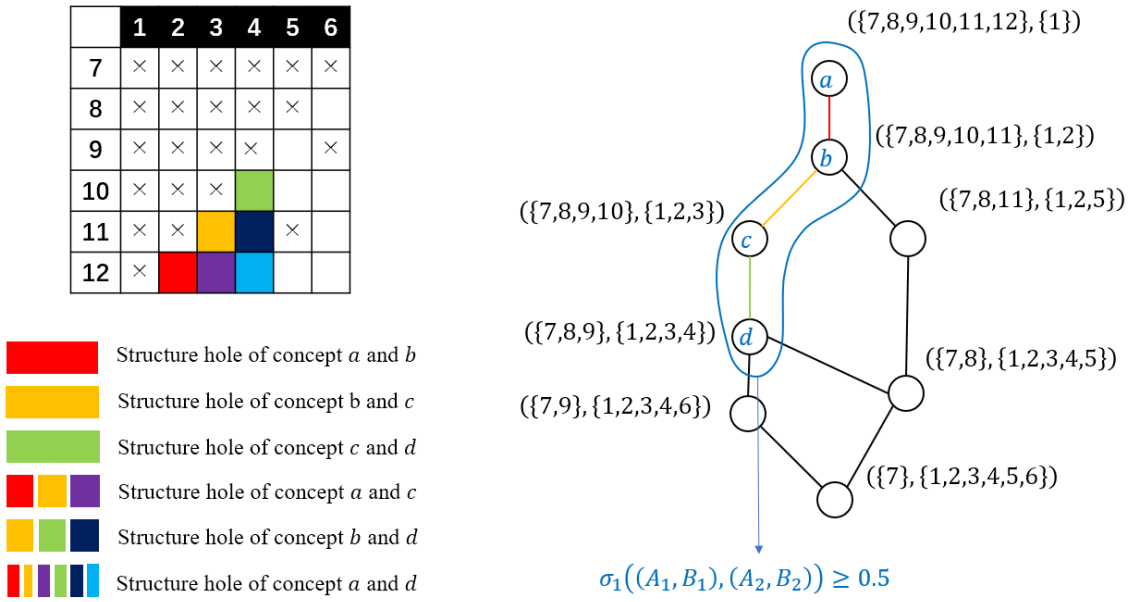


Fig. 4.6 An example of a chain-shaped cluster corresponding to a group of non-trivial overlapping bi-cliques. The structure hole of each pair of concepts is marked with cells in different colors. It is clear that the structure hole of the maximum and minimum concept, *i.e.*, concept a and d contains that of any other pair of concepts from the group.

a proper b. With such a derivation, we can simply enumerate all formal concepts in a depth-first-search mode starting from the minimum concept $(\emptyset^{(1)}, \emptyset^{(1)(2)})$, just like the prototype algorithm. However, with such a procedure, we may still enumerate repeated concepts simply because one concept may have multiple infima. To avoid repetitions, we need to define a topological order for all these concepts. Here we follow the idea of the LCM algorithm to define the predecessor of concept (A,B) as the concept (A_1,B_1) satisfying that $\text{AUG}((A,B),b) = (A_1,B_1)$ and that $(B_1 - B) \cap s(b-1) = \emptyset$ where $s(t)$ is defined as $\bigcup_{1 \le i \le t} \{m_i\}$. In each recursive call of the depth-first-search enumeration process, only when the current concept is found to be the predecessor of the newly generated concept should we call the next recursion. More details of the algorithm can be founded in the pseudocode presented in Algorithm 11.

After enumeration, we are to extract the non-trivial overlapping concepts, *a.k.a.*, non-trivial overlapping maximal bi-clique pairs with the help of the *concept lattice*. As previously analyzed, these bi-clique pairs are clustered into iceberg-shaped and chain-shaped sections in the concept lattice. Since the recursive procedure we introduce before strictly follows the hierarchical order $<$ to traverse the lattice, recording some intermediate parameters during the recursive enumeration procedure allows us to find these clusters after the enumeration

Algorithm 11 The basic DFS procedure for enumerating all concepts.

```
1: procedure ENUMERATE((A,B))
2:     Output (A,B)
3:     for b ← |M| downto 1 do
4:         (A_1,B_1) ← AUG((A,B),b)
5:         if (B_1 − B) ∩ s(b − 1) = ∅ then
6:             Call ENUMERATE((A_1,B_1))
7:         end if
8:     end for
9: end procedure
```

easily. Details can be found in the pseudocode presented in Algorithms 12 and 13. Note that in the pseudocodes, we use some queue-like structures, which are first-in-first-out lists supporting the following operations:

- $Q.push(A)$ adds an element A to the back end of the list.
- $Q.pop()$ removes the element at the front end of the list.
- $Q.front()$ returns the element at the front end of the list.

It can be derived from the code that the recursive ENUMERATE function presented in Algorithm 11 will be called exactly $|C|$ times, where $|C|$ is the number of maximal bi-cliques in the network. After enumeration, we generate C non-trivial overlapping maximal bi-cliques and traverse them in Algorithm 4.3. Hence, the overall complexity of FCA-based negative sample selection is $O(|C|)$, *i.e.*, linear to the number of bi-cliques from the network.

4.3.4 Matrix factorization

In this research, we are mainly to study the effect of our original technique of negative sample selection, so for this step, we only use the most basic form of an MF-based link prediction. That is, the adjacent matrix A of the network is approximated to a product of a $N \times k$ matrix P and a $k \times M$ matrix Q, where K is far smaller than N and M:

$$A = PQ$$

Since the ranks of P and Q are far smaller than that of A, the values of P and Q are not unique. Hence, to compute P and Q, we need to optimize the object function below:

$$\min_{P,Q} |A - PQ|_2$$

Algorithm 12 The recursive procedure for enumerating non-trivial overlapping concepts.

Note The parameter (A,B) represents the concept in process. U is a queue memorizing the current chain-shaped cluster of non-trivial overlapping concepts. *Sim* is a global map-like structure for recording the bounds of clusters. $\mathbb{K} = (G,M,Y)$ is the formal concept. *TList* is a global list of formal concepts.

procedure PREF(b)
    Return $\bigcup_{1 \leq i \leq b} m_b$
end procedure
procedure ENUMERATE($(A,B),U,Sim,\mathbb{K},TList$)
    while $|U| > 0$ and $\sigma_1(U.top(),(A,B)) < \rho$ **do**
        $U.pop()$
    end while
    if $|A| \geq \alpha$ and $|B| \geq \alpha$ **then**
        $U.push((A,B))$
        $TList \leftarrow TList \cup (A,B)$
        if $|U| > 0$ **then**
            $Sim((A,B)) \leftarrow U.front()$
        else
            $Sim((A,B)) \leftarrow (A,B)$
        end if
    end if
    for $b \leftarrow |M|$ downto 1 **do**
        $(A_1,B_1) \leftarrow$ AUG($(A,B),b$)
        if $(B_1 - B) \cap$ PREF($b-1$) $= \emptyset$ and $|A_1| \geq \alpha$ **then**
            Call ENUMERATE($(A_1,B_1),U,Sim,\mathbb{K},TList$)
        end if
    end for
end procedure

Algorithm 13 The main procedure for enumerating non-trivial overlapping concepts.

Input $\mathbb{K} = (G, M, Y)$, the formal context corresponding to the bi-adjacent matrix; α, the size threshold; ρ, the overlapping rate threshold.

Output *List*, the list of preliminary predicted links.

```
UList, DList, List, Y1 ← ∅, ∅, ∅, ∅
Initialize Q as an empty queue.
Initialize UpSim, DownSim as empty maps.
Call ENUMERATE((∅^(1), ∅^(1)(2)), Q, UpSim, (G, M, Y), UList)
for each (g, m) ∈ Y do
    Y1 ← Y1 ∪ (m, g)
end for
Call ENUMERATE((∅^(1), ∅^(1)(2)), Q, DownSim, (M, G, Y1), DList)
for each (A, B) ∈ UList do
    for each (v1, v2) in the structure hole of (A, B) and UpSim((A, B)) do
        List ← List ∪ (v1, v2)
    end for
    for each (v1, v2) in the structure hole of (A, B) and DownSim((A, B)) do
        List ← List ∪ (v1, v2)
    end for
end for
```

The object function is expanded into the following form:

$$\min_{P,Q} \sum_{(v_i,v_j)\neq E_x} (a_{ij} - \vec{p_i}^T \vec{q_{\cdot j}})^2$$

where $\vec{p_i}$ and $\vec{q_{\cdot j}}$ represents the row vector of the i-th row of matrix P and the column vector of the j-th column of matrix Q, correspondingly. It is clear from this form that a_{ij} is the status of a link (v_i, v_j) in the network, and $\vec{p_i}^T \vec{q_{\cdot j}}$ is our confidence score of whether it should be a new link. While this object function is enough to derive and compute the factor matrices, to improve the accuracy, the confidence score is usually added with a *bias* term:

$$b_{ij} = \mu + b_i + b_j$$

where μ is the overall average confidence score of all links, with b_i and b_j being the bias for node v_i and v_j. The bias term is used to force the model to retrieve interactive features between all nodes, instead of the local features of a single node. Besides the bias, the object function is also usually added with a regularization term to prevent over-fitting:

$$\lambda_{ij} = \lambda(b_i^2 + b_j^2 + \|\vec{p_i}|_2 + |\vec{q_{\cdot j}}\|_2)$$

where λ is a hyper-parameter. With these two terms added, the final object function is derived to be:

$$\min_{P,Q} \sum_{(v_i,v_j)\neq E_x} (a_{ij} - \vec{p_i}^T \vec{q_{\cdot j}} - b_{ij})^2 + \lambda_{ij}$$

The function is not convex. However, if we regard either P or Q as constant, it will become a quadratic function and can be optimized using stochastic gradient descending. Hence, to solve this optimization problem, we use a collaborative filtering strategy [55]. That is, the parameter matrix P and Q are updated alternatively in an iterative mode. In each iteration for a training sample a_{ij}, we are to conduct the following four updates:

$$\begin{aligned} b_i &\leftarrow b_i + \gamma(e_{ij} - \lambda b_j) \\ b_i &\leftarrow b_j + \gamma(e_{ij} - \lambda b_i) \\ \vec{p_i} &\leftarrow \vec{p_i} + \gamma(e_{ij}\vec{q_{\cdot j}} - \lambda \vec{p_i}) \\ \vec{q_{\cdot j}} &\leftarrow \vec{q_{\cdot j}} + \gamma(e_{ij}\vec{p_i} - \lambda \vec{q_{\cdot j}}) \end{aligned}$$

where $e_{ij} = a_{ij} - \vec{p_i}^T \vec{q_{\cdot j}} - b_{ij}$ is the predicting score for the sample, and γ is the learning rate, a hyper-parameter controlling the step size of each gradient descent. We repeat the iterations until every training sample is updated at least i times, where i is the pre-determined parameter for the maximum number of iterations.

After the training process is finished, we compute the confidence score for each node pair $(v_i, v_j) \in E_x$ with the aforementioned equation $e_{ij} = a_{ij} - \vec{p_i}^T \vec{q_{\cdot j}} - b_{ij}$. We predict a new link between this node pair if the score exceeds a threshold.

From above, it can be derived that the MF process requires i rounds of updates and each round has about NM update operations. Hence, the time complexity of the MF process is $O(iNM)$, and the overall time complexity of our MF-NSS algorithm is $O(|C|) + O(iNM)$. For comparison, we also list the time complexity of several previous link prediction methods in Table 4.1. Here in the table, D_e refers to the average degree of a node, $|E|$ refers to the number of edges. From the table, we can see that, unlike the previous methods, the time complexity of our method is strongly associated with $|C|$, the number of maximal bi-cliques with enough sizes in the network. Since $|C|$ may grow exponentially as N and M increase, generally, our method is considered slower than the other methods in the list. Nevertheless, in practical cases, we can still make our method finish execution in a reasonable time since we can prevent the value of $|C|$ from overgrowing by tuning the parameter α. In practical, the maximum number of $|C|$ that can make the whole algorithm finish running in a reasonable time (*i.e.*, fewer than 6 hours) is estimated to be 10^5.

Table 4.1 The comparison of the time complexity of different bipartite link prediction methods.

Algorithm	Global or Local	RML or SML	Time Complexity
AA	Local	RML	$O(NMD_{\mathrm{e}})$
JC	Local	RML	$O(NMD_{\mathrm{e}})$
PA	Local	RML	$O(NMD_{\mathrm{e}})$
CN	Local	RML	$O(NMD_{\mathrm{e}})$
RWR	Global	RML	$O(i\|E\|)$
MF	Global	SML	$O(iNM)$
SRNMF	Global	SML	$O(NMD_e)+O(iNM)$
MF-NSS	Global	SML	$O(\|C\|)+O(iNM)$

4.4 The experiments

In this section, we examine our algorithm's effectiveness and performance using two different scenarios of applications. We compare our algorithm with two supervised link prediction methods – raw MF and SRNMF as well as five unsupervised link prediction methods – AA, JC, CN, PA and RWR. All algorithms are written in Python and executed on a Ubuntu 18.04 System with a 2.4GHz CPU and 93.00GB RAM. The codes are available at https://github.com/MF-NSS/MF-NSS.

4.4.1 Link prediction on networks without ground truth for absent links

In this experiment, we study the basic application scenario of link prediction on a network where there is no ground truth for absent links, *a.k.a*, negative samples. Such a scenario is frequently encountered in practical cases, as introduced at this paper's beginning. However, since it is impossible to estimate the performance of the algorithms without ground truth negative samples, in this experiment, we still need to build the input and target networks from datasets with ground truth absent links available and then hide the information from the algorithms to simulate such a scenario. Hence, we choose the MovieLens 25M dataset [43] and HetRec 2011 user-rating dataset [12] to build the networks for this experiment. Both datasets have records of users' ratings of movies from different websites, including IMDb, Rotten Tomatoes, and MovieLens. We process and build bipartite networks from these datasets with the following steps:

- First, we remove the users who rated fewer than t movies and the movies which are rated by fewer than t users from the datasets. Here t is set to 1000 for the MovieLens

dataset and 50 for the HetRec dataset. The remaining users and movies are converted to the two components of the bipartite network, *i.e.*, V_{t1} and V_{t2}.

- Then, we are to build the target network $G_t = (V_{t1}, V_{t2}, E_t = E_t^+ \cup E_t^-)$. For a user u and a movie v, if the user rates 4.5 or higher for the movie, we add (u,v) to E_t^+; if the user rates lower than 4.5 for the movie, we add (u,v) to E_t^-.

- Next, we randomly remove 20% of links from E_t^+ to create E_i, *i.e.*, the link set for the input network. Especially note that in order to simulate the scenario without ground truth for negative samples, here all links in the input network are present links.

After the preprocessing, we feed the input network $G_i = (V_{t1}, V_{t2}, E_i)$ to the algorithms and test if they can predict the target network G_t. For the MovieLens data, we have $V_{t1} = 2675, V_{t2} = 3794$ and $E_t^+ = 398,911$. For the HetRec data, we have $V_{t1} = 1872, V_{t2} = 3261$ and $E_t^+ = 128,657$. Since the input network generation has randomity, we conduct such a network generation five times, run the six algorithms, and collect the average results into Table 4.2. The standard deviations for all scores in the table are below ± 0.001.

Table 4.2 The statistics of the experiment on the MovieLens and HetRec datasets.

	MovieLens		**HetRec**	
Algorithm	**AUC Score**	**AUPR Score**	**AUC Score**	**AUPR Score**
AA	0.619	0.142	0.546	0.160
JC	0.613	0.137	0.536	0.152
PA	0.551	0.107	0.492	0.124
CN	0.620	0.142	0.548	0.161
RWR	0.627	0.132	0.616	0.161
MF	0.779	0.358	0.703	0.291
SRNMF	0.620	0.233	0.621	0.277
MF-NSS	**0.822**	**0.396**	**0.740**	**0.324**

Bold-font numbers represent the highest performance in terms of the corresponding measurement of the corresponding dataset.

From the results, we can see that while the two supervised methods – MF and SRNMF perform better than the other five unsupervised methods on both datasets, our MF-NSS method can still further improve their performance. This suggests that the lack of information on absent links does influence the performance of supervised link prediction, while our technique of negative sample selection can lessen such a negative influence and make it possible to give more accurate predictions with limited information.

4.4.2 Aggressive link prediction on networks where most links are unobserved

In this experiment, we simulate the scenario of making aggressive but reasonable hypotheses based on a small group of observed data. Such a scenario is frequently encountered in many fields like drug development – researchers may start their experiments on those chemicals predicted to be most likely to react with the interested diseases. Motivated by this, we chose the CTD chemical-disease interaction database as the dataset for this experiment. The dataset has records of two types of interactions – the *curated* ones and the *inferred* ones. The curated interactions refer to those with direct evidence published in literature curated by the CTD organization, and the inferred ones are those without direct evidence but can be inferred from external curated sources. These inferred interactions are considered important references for other research in related fields. Hence, if our algorithm can predict these inferred interactions based on the network structural features of the curated interactions only, it suggests that our algorithm may have a great ability to dig deep into the features of these networked interactions and can be used as an alternative method of inference when external curated sources are unavailable. Based on this idea, we make an input network G_i where only the curated interactions are collected into E_i^+ and make a target network G_t where both curated and inferred are collected into E_t^+ and the remaining chemical-disease pairs are collected into E_t^-. After processing, we have $V_{t1} = 10225, V_{t2} = 3283, E_i^+ = 103,845$ and $E_t^+ = 1,965,562$. We run the six algorithms five times and collect the average AUC scores and AUPR scores into Table 4.3. The standard deviations of all scores in the table are around ± 0.001 to ± 0.002.

Table 4.3 The statistics of the experiment on CTD chemical-disease database.

Algorithm	AUC Score	AUPR Score
AA	0.373	0.046
JC	0.382	0.049
PA	0.503	0.029
CN	0.388	0.052
RWR	0.500	0.021
MF	0.688	0.076
SRNMF	0.731	0.118
MF-NSS	**0.846**	**0.276**

Bold-font numbers represent the highest performance in terms of the corresponding measurement.

As can be seen in Table 4.3, Our MF-NSS method significantly improves both the AUC score and the AUPR score compared to all other methods. Although both the raw MF method and the SRNMF method have improved the AUC score compared to all four unsupervised methods, their AUPR scores are close to the baseline. According to previous studies [22], the AUPR score gives a more focused estimation of a model's ability to recognize positive samples, so the low AUPR scores imply that these two methods are insensitive to present links. We analyze this because the input network in this experiment is only about 10% as dense as the networks used in the last experiment and thus has many more non-concrete absent links. Without a specially designed pre-processing procedure, these two methods will treat all these non-concrete absent links as reliable negative samples and as a result, they should have a high tendency to predict an absent link between most node pairs. On the other hand, our MF-NSS method can still make accurate predictions, thanks to the technique of negative sample selection. This perfectly shows the effectiveness of our NSS procedure and that our algorithm is suitable for such an application scenario. Also, we found that in this experiment, RWR has the lowest AUPR score among all methods that we tested, even compared to the four local methods. A similar conclusion was also mentioned in [120] that unsupervised methods may not work well in protein-protein networks since they have different link formation mechanisms than the more-common social networks or user-movie rating networks. This indicated that global methods do not always outperform local methods, but that supervised methods generally perform better than unsupervised methods.

4.4.3 Studies on the performance of negative sample selection

In this subsection, we discuss the factors that affect the performance of the negative sample selection step. As mentioned above, in this step, we first use an FCA-based preliminary link prediction method to mark out some node pairs that are least likely to be negative samples and then randomly select a certain percentage of node pairs as negative samples from the rest of the unobserved part of the network. It can be easily derived that the parameters which control the output of the FCA-based link prediction method and the sample rate, *i.e.*, the determined percentage of the selected negative samples, should influence the final results most significantly. Hence, we design the following experiment to study how it will influence the results: First, besides the FCA-based negative sample selection procedure, we implement another procedure that completely randomly selects negative samples. Then, we run both procedures with different sample rates, feed their selected training samples into the MF procedure, and plot the final results into Fig 4.7. From the results, it is clear that we may find some best sample rates contributing to the best AUC and AUPR scores – either higher or lower sample rates will result in lower scores. Also, we discover that when the sample rate

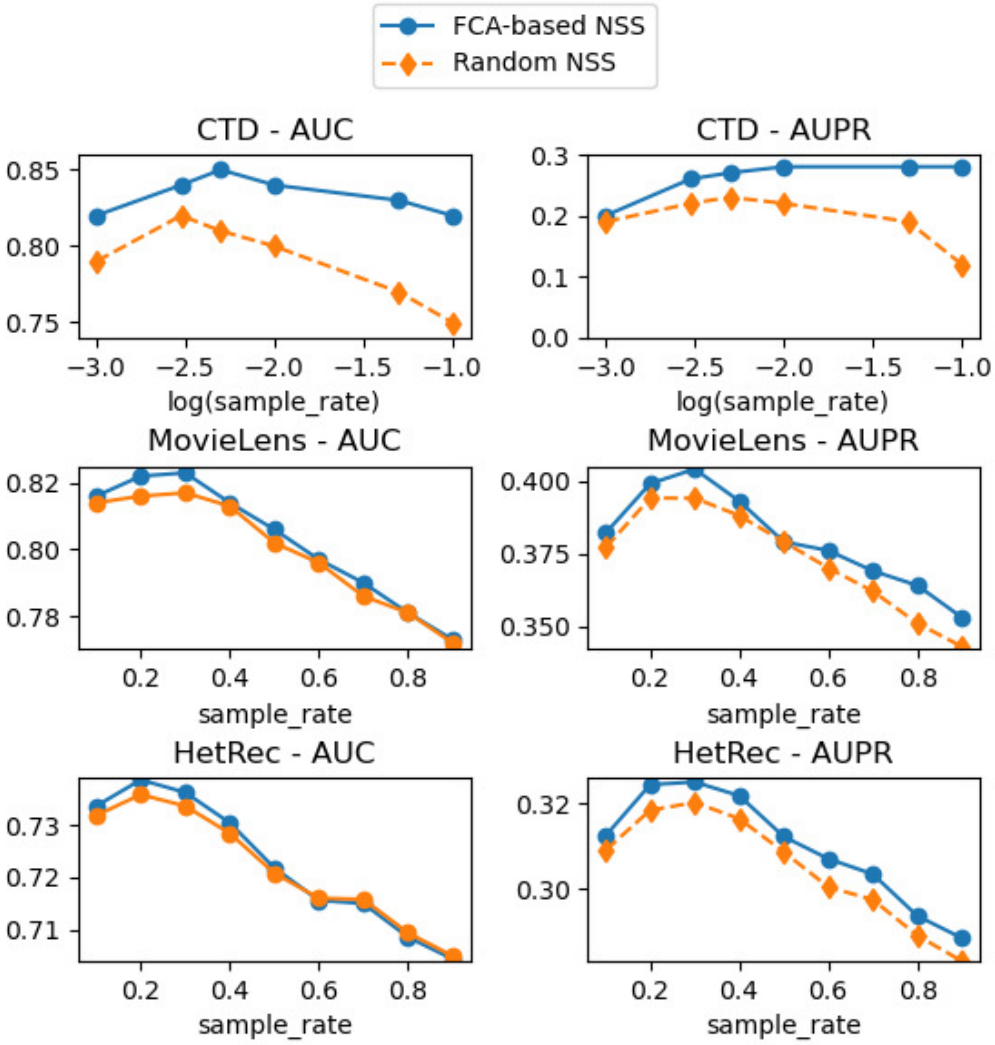


Fig. 4.7 The AUC and AUPR scores with different sample rates.

is set to be these best values, both the random negative sample selection procedure and our FCA-based selection procedure can contribute to their highest AUC and AUPR scores, while the peak scores achieved by our FCA-based procedure are higher than a random selection procedure over all three datasets. This shows that our FCA-based negative sample selection procedure is effective and gives us the idea of using a random selection procedure to estimate the best sample rates before running the joint system. Furthermore, we have also found that the AUC and AUPR scores of our FCA-based negative sample selection have the largest margin over that of the random selection procedure in the task of aggressive link prediction. We analyze that it is because, in this task, there are many more positive samples in the test set, which means that a random negative sample selection will have a high probability of selecting a potential new link as a negative sample. Our FCA-based link prediction method, however, can prevent such a case from happening and thus can keep good performance in this scenario. On the other hand, in the other application scenario of link prediction without ground truth for negative samples, if there is no urgent need for high performance, it is also recommended to apply a random negative sample selection, which will slightly reduce the AUC and AUPR scores but still has a very good performance compared to a raw MF procedure or other unsupervised methods.

4.5 Conclusion of the chapter

We proposed an FCA4SML method called MF-NSS for the task of link prediction on bipartite networks. The method combines the traditional MF-based link prediction procedure with a unique FCA-based preliminary negative sample selection technique. We studied the application of this method to two typical scenarios – making aggressive predictions when most of the network is unobserved and making link predictions on a network where no ground truth for absent links is available. In the former scenario, the technique significantly improves the AUC and AUPR scores, showing the possibility of MF-NSS being used as an alternate chemical-disease or gene-disease inference method. In the latter scenario, our method can lessen the affection of excessive negative samples to reach the best performance compared to all other methods. The good performance of our method in both scenarios shows that even if we do not directly process the concept lattice with and SML method, we can still apply other strategies like our proposed negative sample selection technique to make the SML method fully utilize the information provided by FCA.

As our future work, we consider extending our method so that it can be further applied to heterogeneous networks, that is, networks where the links have different types. Fortunately, there has already been research on the three-dimensional extensions of formal concept

analysis and matrix factorization, which will provide us with crucial theoretical basis and show us the possibility of our planned extended method. Also, we plan to develop more ways for negative sample selection and combine this technique with other SML methods, to see if it can boost the performance of these methods as well.

Chapter 5

An FCA4SML method which effectively captures both types of information in concept lattices

5.1 Introduction

In this chapter, we propose BERT4FCA, an FCA4SML method which utilizes a BERT-like Transformer encoder framework to efficiently capture both types of the information from the concept lattice and utilize them for downstream machine learning tasks. We propose this as our solution for the third issue that makes it hard to develop a working FCA4SML method, that is, the information within the concept lattice is complicated and thus hard to be learned by SML methods.

As introduced in Chapter 1, there are two types of information in the concept lattice – one is the relations between objects/attributes within the formal concept, and the other one is the order relation between different formal concepts. To the furthest of our knowledge, all previous FCA4SML methods, can only capture and utilize the first type of information, while we consider the two types of information equally important and neither one should be left out. For example, in the context shown in the left panel of Fig. 5.1, let us use (A_1, B_1) to denote the concept $(\{g_1, g_2\}, \{m_1, m_2, m_3\})$ and (A_2, B_2) to denote the concept $(\{g_1\}, \{m_1, m_2, m_3, m_4, m_5\})$. If we only consider the information within the concept (A_1, B_1) and (A_2, B_2), we may only derive conclusions like that "the objects in A_1 are strongly related to the attributes in B_1". However, from their order relation in the concept lattice, we can know that (A_1, B_1) is the direct neighbor of (A_2, B_2). This implies that there does not exist an object $g \in A_2 - A_1$ such that $A_{\mathbb{K}}(\{g\})$, *i.e.*, the attribute set of g, satisfies that $B_1 \subseteq A_{\mathbb{K}}(\{g\}) \subseteq B_2$.

In other words, for objects g that already has attributes m_1,m_2 and m_3, attributes m_4 and m_5 will be equivalent because all these objects should have both attributes. Such derived information can provide important hints on machine learning tasks like link prediction or node classification, so we consider it necessary to develop an SML method that can effectively capture both types of information.

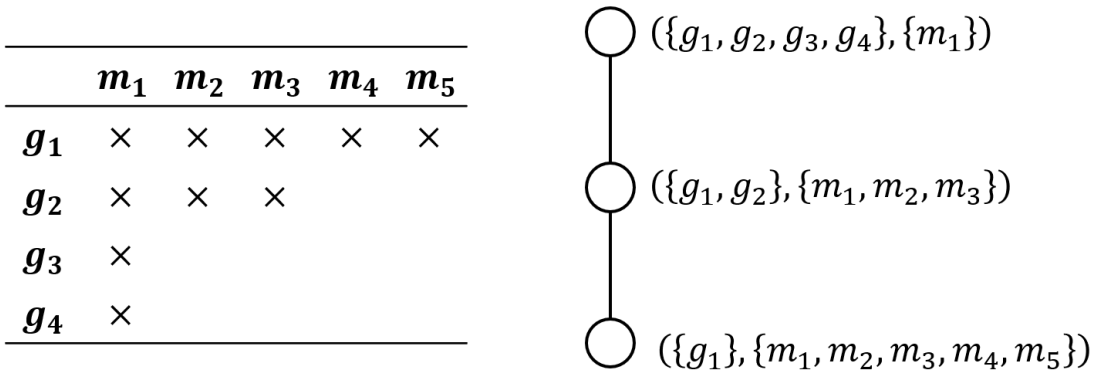


Fig. 5.1 Left: A sample context. Right: The concept lattice corresponding to the context to the left.

To achieve the goal, we propose a novel FCA4FML method named *BERT4FCA*, which processes the concept lattice with a popular method in natural language processing method called *BERT* or *Bidirectional Encoder Representations from Transformers* [107]. BERT is a method for training language models utilizing the *Transformer* encoder architecture [107]. BERT first pre-trains a large model on unlabeled free text to learn the co-relations between words and sentences; then, it fine-tunes the model on a small labeled dataset to fit a target downstream task. We chose BERT because we found that the both types of information provided by FCA, that is, the formal concepts and their hierarchical relations, share similarities with the input data that BERT takes. Furthermore, after we pre-training a model which captures these two types of information, by fine-tuning the pre-trained model, we many be able to make them fit any types of downstream tasks. In other words, unlike the method introduced in the last chapter which is task-specified, the method introduced in this chapter is generally applicable to different downstream tasks.

To verify the practical applicability and the versatility of our method, in this chapter, we focus on the application of our method on two different tasks. The first task is link prediction on bipartite network, which has been explicitly introduced and discussed in the last chapter.

The second task is *hyper-link prediction on bipartite network*, which focuses on predicting the relations between nodes from the same node sets of a bipartite network. For two nodes from the same set that are not connected to the same node in the other set, it predicts whether there should be an unobserved node in the other set that is connected to both nodes. For example, in an author-paper network, for two authors that do not have a co-authorship, it predicts if they will have a new co-authorship in the future [13, 28]. Like the task of bipartite link prediction studied in the last chapter, this task have also attracted increasing attention for its high practical values [32, 73, 115].

The rest part of this chapter is organized as follows. In Section 5.2, we start with some preliminaries, including BERT and its relation with FCA. In Section 5.3, we present the problem formulation of this study and the related work on the two tasks studied in this chapter. In Section 5.4, we introduce and analyze our method BERT4FCA. In Section 5.5, we describe our experiments on five real-world datasets and discuss the results. Finally, in Section 5.6, we draw a conclusion and discuss our plans for future work related to this sub-topic.

5.2 Preliminaries

5.2.1 BERT

BERT, short for *Bidirectional Encoder Representations from Transformers* [24], is a method for training a language model via the *Transformer* architecture [107]. It works in a "*pre-train* first and *fine-tune* next" mode. First, it pre-trains a large language model with two general tasks on a large amount of unlabeled free text. Then, it fine-tunes the pre-trained model with a specific downstream task on a small labeled dataset.

The two general tasks used in the pre-training phase are *mask language model* (MLM) and *next sentence prediction* (NSP), which are defined as follows.

MLM: MLM is the task that predicts the full sentence from a sentence where some words are randomly masked with a special token "[MASK]". For example, if we have a sentence like "the quick brown fox jumps over the lazy dog", the model will take a masked version of the sentence like "the [MASK] brown fox jumps [MASK] the lazy dog" as input and the original sentence as target for output. The task helps the model learn the co-occurrence relationship between words in the same sentence [24].

NSP: NSP is the task that predicts whether two sentences are the subsequent sentences or not. The model takes a sentence pair (A, B) as input and is expected to output TRUE if B

is the sentence directly following A in the corpus, or FALSE otherwise. The task helps the model to understand the relationships between sentences.

After pre-training, the values of the weight matrices of the pre-trained model are used as initial values for that of the fine-tune model and updated in the fine-tuning phase [29]. This is possible because pre-training and fine-tuning should use exactly the same network architecture except for the output layer. The downstream task in the fine-tuning phase should be the final target task. Hence, after fine-tuning, we get the final model which can be directly used for our target task.

5.2.2 BERT and FCA

As mentioned above, BERT is a method for training language models, which takes sentences in natural languages as inputs. Although formal concepts are not sentences in natural languages, they do share some similarities – If we regard objects and attributes as words, then the extents and intents of a concept can be regarded as sentences in a language that has an order-free syntax. With such similarities, we may expect that the pre-training phase of BERT, which was originally designed for learning the features of words and sentences, can also be used for learning the features of objects, attributes, and formal concepts from a formal context and its corresponding concept lattice. We may also expect that link prediction and hyper-link predictions can be suitable downstream tasks after such a pre-training because the information of nodes (*a.k.a.* objects and attributes), and maximal bi-cliques (*a.k.a.* formal concepts) learned in the pre-training phase is helpful in increasing the accuracy of predictions.

Note that in most natural languages, the syntax is not order-free, indicating that different word orderings usually have completely different meanings. Hence, BERT has a special mechanism called *position embedding* for capturing the order of words in the input sentence. However, since the extents and intents of formal concepts are unordered, there is no need to keep track of the order. Hence, in our method, this mechanism is removed from BERT.

5.3 Problem formulation and related work

The chapter studies two different tasks on bipartite networks – the first task is *hyper-link prediction on bipartite networks*, which predicts the missing or unknown relation between two nodes from the same node set of a bipartite network, and the second task is *bipartite link prediction*, which predicts the missing or unknown relation between two nodes from different node sets. As analyzed above, a bipartite network is equivalent to a formal context, and the two node sets are equivalent to the object set and the attribute set. Hence, for ease of

understanding, we hereby name the two tasks as the *object-object task* or the *O-O task* and the *object-attribute task* or the *O-A task*. The two node sets of a bipartite network are also directly called the *objects* and *attributes* in the rest part of the paper.

The formal definitions of the two tasks are defined as follows.

O-O task: Given an original network $C = (U, V, E)$ and a target network $C' = (U, V', E')$, the O-O task aims to predict if a group of nodes $G = \{u_1, u_2, \cdots, u_{|G|}\} \subseteq U$ that does not have *object-object links* in C, should have object-object links in C'. For a group of nodes $G \subseteq U$, they are considered to have *object-object links* in a network (U, V, E) if $\exists v \in V$ such that $\forall u \in G$, $(u, v) \in E$.

O-A task: Given an original network $C = (U, V, E)$ and a target network $C' = (U, V, E')$, the O-A task aims to predict if two nodes $u \in U$ and $v \in V$ that does not have an *object-attribute link* in C, should have an object-attribute link in C'. For two nodes $u \in U$ and $v \in V$, they are considered to have an *object-attribute link* in a network (U, V, E) if $(u, v) \in E$.

5.3.1 Related work

For the O-O task, the notable recently proposed methods are the two FCA4SML methods briefly introduced in Chapter 1. In [28], the authors proposed an embedding-based method called *object2vec* to embed objects into vectors using the information from the formal concepts. Dually, they have also proposed *attribute2vec* for embedding attributes into vectors, which uses exactly the same mechanism as object2vec. It has two embedding models, *object2vec-CBoW* and *object2vec-SG*, both are derived from *Word2Vec* [75, 76]. Object2vec-CBoW, based on the *continuous-bag-of-words* model from Word2Vec, predicts a target object using objects around it within the same extent; object2vec-SG, based on the *skip-gram* model from Word2Vec, uses an object to predict other objects in the same extent. They conducted experiments on the O-O task on an author-publication network and demonstrated good performance. In [69], the authors proposed another embedding method called *Bag of Attributes (BoA)*. Their method trains a more complicated embedding model using *Bidirectional Long Short-Term Memory* (Bi-LSTM) [39] and *Variational Autoencoder* (VAE) [53] on formal contexts. They conducted experiments on the O-O task on the same datasets as object2vec, and the results are similar to those obtained with object2Vec.

In [121], the authors proposed a rule-based method for the O-A task by analyzing the overlapping formal concepts from a formal context. If the ratio of the overlapped part exceeds a predefined threshold, they are considered parts of the same large formal concept, so all the missing links in the non-overlapped part *a.k.a. the structure hole* [63] are predicted as present links. It presents an interesting approach to bipartite link prediction but only has a limited performance.

Besides these two FCA4SML methods, other recently proposed O-O methods include *HPRA* [61] and *HyperTensor-EVD*[72]. HPRA is an RML method that applies an algorithm that computes the *Hyper Resource Allocation* (HRA) scores to determine if an object should be included in an object group where the objects are predicted to be related. HyperTensor-EVD is an SML method that computes the prediction score of a potential O-O link from the eigenvectors of the tensor-based representation of the hyper-network, which can be built from the O-O links of the bipartite network [72].

As analyzed in Chapter 1, the two FCA4SML methods only utilize the information within the concepts, but not the information on the hierarchical order of the concepts. Furthermore, although these two methods have shown a good performance, these are still outperformed by other non-FCA-based methods like HPRA, so we consider it is worth trying to design a better FCA4SML framework to capture the omitted information on the hierarchical order of the concepts to see if it can contribute to better performance results.

For the O-A task, besides the matrix factorization methods and the classic RML methods introduced in the last chapter, recently proposed methods also include *SBGNN*, a deep-learning-based method which applies a special network structure called *Signed Bipartite Graph Neural Network* to capture and learn the information in the observed part of the network and thus predict the unobserved part.

5.4 The proposed method

To address the limitations of the previous FCA4RML and FCA4SML methods for these tasks, we propose a novel method named *BERT4FCA*, which is designed to better learn the information of concept lattices and can conduct both the O-O task and the O-A task. We name our method *BERT4FCA* because it provides a general framework for using BERT to learn and utilize the information of concept lattices, so it is expected to be generally applicable to all tasks related to the context. Although in this chapter, we only discuss its application in the two tasks, we plan to study the possibility of applying our method to other tasks in the future.

Similar to all previous FCA4RML and FCA4SML methods for these tasks, the objective of our method is to learn information from concept lattices and use it to make two link prediction tasks. However, the information our method aims to learn not only includes the extent and intent of a formal concept, but also includes the neighboring relations between formal concepts. Note that here we choose to learn the *neighboring relations* instead of the general *order relations* because, to reconstruct the concept lattice, the neighboring relations are enough [37].

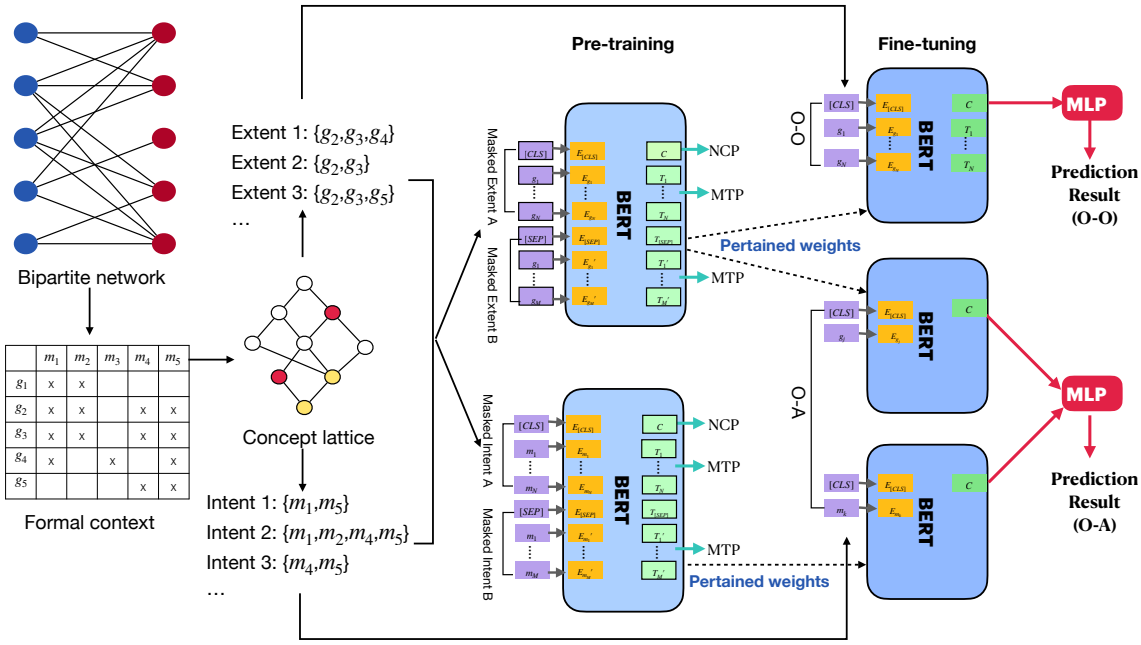


Fig. 5.2 An overview of the working flow of our method.

Our method consists of 4 steps: *data preparation with FCA*, *input tokenization*, *BERT pre-training*, and *BERT fine-tuning*. An overview of the workflow of our method is shown in Fig 5.2.

Data Preparation: In this step, we convert the bipartite network into a formal context, extract all formal concepts, and construct the concept lattice. Then, we extract the neighboring relations between concepts from the concept lattice. We use the Z-TCA algorithm introduced in Chapter 2 for extracting all formal concepts from a formal context, and use a *topological sorting* algorithm to extract neighboring relations from the concept lattice. The details of the algorithm are presented in Algorithm 14. After this step, we will obtain all extents and intents and neighboring concepts in the concept lattice.

Input Tokenization: In this step, the objects and attributes are tokenized into one-hot vectors. These one-hot vectors are further converted into dense vectors through the *input embedding* so that they can be processed by BERT.

The input embedding is the sum of two parts: *tokenization embedding* and *segment embedding*. Tokenization embedding is the general-sense embedding that uses a full-connect layer to embed the tokenized one-hot vectors into a dense vector space. Segment embedding is used to embed the tokens in the segment-info sequence into the vector space of the same dimension as the tokenization embedding. In BERT, sometimes the training sample may be a concatenation of two different sequences (details will be introduced later). In this case, the segment-info sequence is generated to distinguish the two different sequences within the

Algorithm 14 Get all neighboring relations using topological sorting.

Input A formal context $\mathbb{K} = (U, V, E)$ and its concept lattice $\underline{\mathfrak{B}}(\mathbb{K})$.

Output $\{\mathrm{N}(C)\}$, a list of the lower neighbors of concepts. Here $\mathrm{N}(C)$ represents the lower neighbor of $C \in (\underline{\mathfrak{B}}(\mathbb{K}) - (\emptyset, \emptyset''))$.

for $C \in \underline{\mathfrak{B}}(\mathbb{K})$ **do**
    $\mathrm{D}(C) \leftarrow |\{C_1 \in \underline{\mathfrak{B}}(\mathbb{K}) \mid C_1 < C\}|$
end for
Create Q as an empty queue.
for $C \in \underline{\mathfrak{B}}(\mathbb{K})$ **do**
    if $\mathrm{D}(C) = 0$ **then**
        Push C into the back of Q.
    end if
end for
while Q is not empty **do**
    Fetch C from the front of Q and pop it.
    for $C_1 \in \underline{\mathfrak{B}}(\mathbb{K})$ such that $C < C_1$ **do**
        $\mathrm{D}(C_1) \leftarrow \mathrm{D}(C_1) - 1$
        if $\mathrm{D}(C_1) = 0$ **then**
            $\mathrm{N}(C_1) \leftarrow C$
            Push C_1 into the back of Q.
        end if
    end for
end while

same training sample. For example, in a training sample, the input is a sequence of seven tokens, where the first four tokens belong to the first sequence, and the last three tokens belong to the second sequence. Then, the segment-info sequence of this training sample should be $(0,0,0,0,1,1,1)$.

Note that, as mentioned above, in the original framework of BERT, there is another input embedding called position embedding used for learning the order of words in a sentence. In BERT4FCA, however, it is removed because the input sequences are all unordered.

BERT Pre-training: In this step, we use the BERT framework to pre-train two models – the *object model* and the *attribute model* – on all extents and intents, correspondingly. Here, we only introduce the pre-training process of the object model for example. The input of a training sample of the pre-training of the object model is a pair of sequences, each representing an extent. Since the lengths of extents may be different, we pad short extents to make all extents have the same length with a special token "[PAD]". We also add a special token, "[SEP]", between two sequences in order to separate them.

The model is trained with two tasks: *masked token prediction* (MTP) and *neighboring concepts prediction* (NCP), derived from the MLM and NSP tasks in the original version of BERT, correspondingly.

MTP is the task that helps the model learn the co-occurrence relationships between objects within the same extent. In the task, we randomly select a certain percentage of objects in both extents to be masked. For each object to be masked, we replace them into a special token "[MASK]" with 80% probability, or replace it with a random object with 10% probability, or keep it unchanged with 10% probability. Then, the model takes the masked pair of extents as input and the unmasked pair of extents as the target for output. That is, it is trained to predict the masked objects in the extents.

NCP is the task that helps the model learn the neighboring relations between formal concepts in the concept lattice. In the task, the model takes a pair of extents as input and is expected to output TRUE if the pair of extents corresponds to a pair of formal concepts that have neighboring relations or output FALSE otherwise. Note that in this task, clearly, the number of negative samples is much larger than the positive samples. To get a balanced training set, we randomly select a small portion of negative samples and keep the number of positive and negative samples to be the same.

The two tasks are trained simultaneously – the training loss is the sum of the losses of the two tasks. After pre-training, the pre-trained models are expected to have captured and stored information on the concept lattice, including relationships between objects and formal concepts.

BERT Fine-tuning: In this step, we fine-tune the pre-trained object model and attribute model to make them fit our target tasks.

For the O-O task, we fine-tune the pre-trained object model with the training samples generated from the original bipartite network (U,V,E). The training samples are generated with the following steps. First, we determine l_m, the maximal length of a group of objects we want to predict. Then, we enumerate all object subsets $U_1 \subseteq U$ such that $|U_1| \leq l_m$. For each U_1, we create a training sample that takes the tokenized and padded sequence of U_1 as input, and the label for the sample is set to TRUE if the objects in U_1 have object-object links and FALSE otherwise. After generation, the training samples are fed into a network that has the same structure as the network used in the pre-training step, except that in the last layer, the hidden states are not fed into the output layer used for the two pre-training tasks, but are fed into an output layer specified to the O-O task. Suppose the final output of the fine-tuning network for the O-O task is $P_{\mathrm{O-O}}$; the last hidden state of the basic BERT network, *i.e.*, the "[CLS]" representation [24] is $h_{\mathrm{L}}^{[\mathrm{CLS}]}$, then the O-O-task-specified output

layer can be described as follows:

$$P_{\mathrm{O-O}} = \sigma(\mathrm{ReLU}(h_{\mathrm{L}}^{[\mathrm{CLS}]} W_{\mathrm{CLS}})W), \tag{5.1}$$

where $\sigma(\cdot)$ is the sigmoid function; $\mathrm{ReLU}(\cdot)$ is the *Rectified Linear Unit* (ReLU) [98]; $W_{[\mathrm{CLS}]}$ and W are weight matrices.

For the O-A task, we fine-tune both the pre-trained object model and the attribute model together with training samples generated from the original bipartite network. The training samples are generated with the following steps. We first enumerate all objects $u \in U$ and attributes $v \in V$. For each pair of (u, v), we create a training sample that takes the tokens of u and v as input, and the label for the sample is set to TRUE if $(u, v) \in E$ and FALSE otherwise. After generation, the tokenized object and tokenized attribute are fed into two separate BERT networks – except for the last output layer, the first network has the same structure as the network used in pre-training the object model, and the second network has the same structure as that used in pre-training the attribute model. The last hidden states of both networks are concatenated and fed through a single output layer specified for the O-A task. Suppose the final output of the fine-tuning network for the O-A task is $P_{\mathrm{O-A}}$; the last hidden states of the two BERT networks are $h_{\mathrm{L1}}^{[\mathrm{CLS}]}$ and $h_{\mathrm{L2}}^{[\mathrm{CLS}]}$, then the O-A-task-specified output layer can be described as follows:

$$P_{\mathrm{O-A}} = \sigma(\mathrm{ReLU}((h_{\mathrm{L1}}^{[\mathrm{CLS}]} \frown h_{\mathrm{L2}}^{[\mathrm{CLS}]}) W_{\mathrm{CLS}})W), \tag{5.2}$$

where $a \frown b$ means the concatenation of vectors a and b.

Above is the whole workflow of our method. From above, we can see that our method has three advantages. First, in the pre-training step, our method can learn more information from concept lattices compared to the previous FCA4SML like object2vec. As shown in Fig 5.3, in object2vec, when embedding an object, it uses only the information of a small set of objects within the same extent. In the pre-training step of our method, however, we learn the feature of an object using the information from all objects in the entire extent as well as the objects in the extents of its neighbor concepts in concept lattice.

Second, our method can directly extract useful features for link prediction from the formal context, in addition to learning from the concept lattice. During fine-tuning step, we train the model with samples generated from the formal context, which enables the model to learn directly from formal contexts. Therefore, even in datasets where the concept lattice cannot provide much useful information for link prediction, our method can still learn from the formal contexts for link prediction. In contrast, previous FCA4SML methods do not

possess the capability of directly learning from the formal context. Additionally, previous non-FCA-based methods cannot make use of bi-cliques for link prediction.

Third, our method works in a pre-train first and fine-tune next mode that first pre-trains two large models and then fine-tunes them on various downstream tasks related to the formal context, meaning that if we wish to conduct new tasks on a formal context where we have pre-trained the object model and attribute model, we can skip the first three steps and directly conduct the final fine-tuning step. Previous methods, however, may need to re-train the models when coming to a new task.

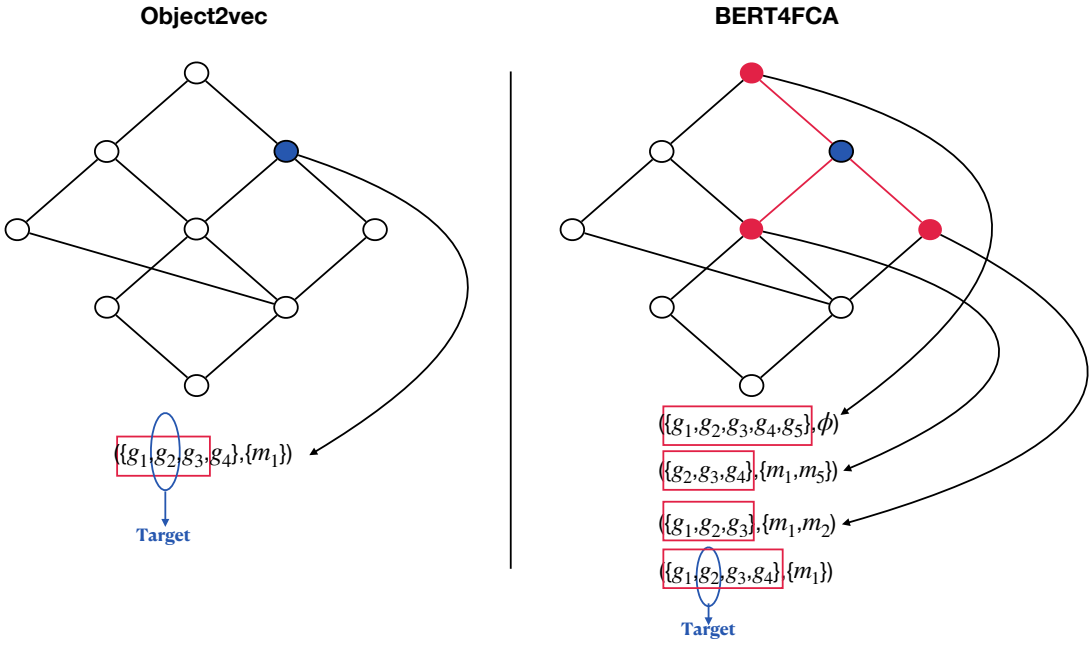


Fig. 5.3 The comparison of how much information from a concept lattice is learned and used when predicting an object by two methods, with object2vec shown on the left and BERT4FCA shown on the right. The target object to be predicted is circled in blue. The information used for predicting the object is shown in red.

5.5 The experiments

5.5.1 Datasets

We conduct experiments on five real-world datasets: *ICFCA*, *BMS-POS*, *Keyword-Paper*, *Review* and *iJO1366*. All datasets are used for both the O-O task and the O-A task. We depict the features of these datasets in Table 5.1. A detailed description of each follows.

Table 5.1 The features of the datasets.

Dataset	Task	Input/Target	Objects	Attributes	Edges	Concepts
ICFCA	O-O	Input	334	12614	13399	775
		Target	334	12614	15980	844
	O-A	Input	351	12614	14445	878
		Target	351	12614	16049	922
BMS-POS	O-O	Input	468	1946	7376	7791
		Target	468	1946	8085	10235
	O-A	Input	468	1946	7376	7791
		Target	468	1946	8085	10235
Keyword-Paper	O-O	Input	162	5640	7274	1610
		Target	162	5640	8308	2049
	O-A	Input	162	5206	7648	1713
		Target	162	5206	7907	2046
Review	O-O	Input	181	340	420	262
		Target	181	340	465	281
	O-A	Input	181	340	420	262
		Target	181	340	465	281
iJO1366	O-O	Input	1805	2583	9231	5185
		Target	1805	2583	10184	5595
	O-A	Input	1805	2583	9231	5185
		Target	1805	2583	10184	5595

ICFCA: The ICFCA dataset is an author-paper network provided by [28] – the objects represent the authors, the attributes represent the publications, and each edge (*a.k.a.* relation) represents the author is in the author list of the publication. This dataset is generated from *Digital Bibliography & Library Project* (DBLP) dump on 1st Aug 2019 which is available at https://dblp.uni-trier.de/xml/.

For the O-O task, we are to simulate the practical case in which we wish to predict future co-authorships or seek potential co-authors from an author-paper network at a certain time point. Hence, we generate a history network from the full network as the input network for this task by removing the authors, publications, and author-paper edges after 1st Jan 2016; we generate the current network as the target network for this task from the full network by removing authors who had no publication before 31st Dec 2015 and their corresponding edges.

For the O-A task, we are to simulate the practical case that some parts of the network are missing, and we wish to use the known edges in the network to predict the potentially missing edges. We generate the input network for this task from the full network by randomly removing 10% of author-paper edges; we use the full network as the target network for this task.

BMS-POS: The BMS-POS dataset is a product purchased transactions network provided by KDD Cup 2000 – the objects represent the products, the attributes represent purchasing transactions, and an edge represents that the product is bought in a certain purchasing transaction. The original data is very large, so in this research, we only select the first 1946 transactions. The dataset is available at https://kdd.org/kdd-cup/view/kdd-cup-2000.

For the O-O task, we are to simulate the practical case in which we wish to predict two products will be likely to be bought by the same customer. Hence, we generate the input network from the full network by randomly removing 10% of product-transaction edges; we use the full network as the target network for this task.

For the O-A task, we are to simulate the same practical case as the ICFCA dataset. We use the same input network and target network as those used in the O-O task of this dataset.

Keyword-Paper: The Keyword-Paper dataset is an original dataset generated by us – the objects represent the keywords, the attributes represent the publications, and each edge represents the paper has the keyword. It is generated from the DBLP dump on 31st Jan 2023. From the dump, we select the top 162 most frequent keywords and all 5640 publications after 1st Jan 2010 to create the keyword-paper network.

For the O-O task, we are to simulate the practical case in which we wish to predict potentially related keywords, which may give inspiration for new research. For example, while “BERT” and “FCA” were never used as keywords of the same paper before, if they are

predicted to be related, researchers may get inspired and draft a new study similar to ours. The generation of the input network and target network are similar to that of the ICFCA dataset, and the date for the history network is also set to 31st Dec 2015.

For the O-A task, we are to simulate the same practical case as the previous two datasets. We also use the same way to generate the input and target networks as the way we used in the O-A tasks of the previous two datasets. Note that in this dataset, after removing 10% of edges, some attributes will have no edge connecting to them, so the number of attributes in the networks used for the O-A task is smaller than that used for the O-O task.

Review: The Review dataset is a peer review data from a top computer science conference. It is collected from [23] and previously used as a test dataset in [45]. The objects represent reviewers; the attributes represent manuscripts; and each edge represent that the reviewer suggests accepting the manuscript.

For both the O-O and the O-A tasks, we are to simulate the practical cases similar to those of the BMS-POS dataset. Hence, the input networks and the target networks are also generated using the same procedures as for the BMS-POS dataset.

iJO1366: The iJO1366 dataset is a metabolite-reaction network processed from a genome scale metabolic network of the Escherichia coli. It is collected from the BiGG dataset [52] and previously used as a test dataset in [116]. We extract the metabolites as objects, and the reactions as attributes, and each edge represents that the metabolite participates in the reaction.

For both the O-O and the O-A tasks, we are to simulate the practical cases similar to those of the BMS-POS dataset. Hence, the input networks and the target networks are also generated using the same procedures as for the BMS-POS dataset.

5.5.2 Generation of labeled training and test samples

For each dataset, we have generated an input network $C = (U, V, E)$ and the target network $C' = (U, V, E')$. We are then to generate labeled training and test samples from the two networks with the following procedure.

The O-O task

For the training samples, we first enumerate all object-object links in C which contains no more than ε_p objects, with ε_p being a pre-determined threshold. These links are labeled as positive training samples. That is, we are to enumerate every object group $G \subseteq U$ satisfying that $|G| \leq \varepsilon_p$ and $\exists v \in V$ such that $\forall u \in G$, $(u, v) \in E$. Then, we are to randomly choose the same number of negative training samples samples as the positive training samples, with

each negative sample being an object-object link which has no more than ε_p objects and does exists in C. Formally, each negative sample should be an object group $G \subseteq U$ such that $|G| \leq \varepsilon_p$ and $\nexists v \in V$ such that $\forall u \in G$, $(u,v) \in E$. The proportion of object-object links with different sizes in the set of negative training samples is kept the same as that in the set of positive training samples. That is, suppose the set of all positive training samples is T_1, and the set of all negative training samples is T_2, we should have $|T_1| = |T_2|$ and for each $i = 2,3,\cdots,\varepsilon_p$, we should have $|\{G \in T_1 \mid |G| = i\}| = |\{G' \in T_2 \mid |G'| = i\}|$.

For the test samples, we enumerate all object-object links in C' which contains no more than ε_p nodes and **does not appear in** C. These links are labeled as positive test samples. That is, we are to extract every object group $G \subseteq U$ satisfying that $|G| \leq \varepsilon_p$ and $\exists v \in V$ such that $\forall u \in G$, $(u,v) \in E' - E$. Then, we are to randomly choose the same number of negative test samples samples as the positive test samples, with each negative test sample being an object-object link which has no more than ε_p objects and does exists in C'. Formally, each negative test sample is a group $G \subseteq U$ such that $|G| \leq \varepsilon_p$ and $\nexists v \in V$ such that $\forall u \in G$, $(u,v) \in E'$. The proportion of object-object links with different sizes in the set of negative test samples is also kept the same as that in the set of positive test samples.

In this research, the value of ε_p is set to 5 for the ICFCA and Review datasets; it is set to 3 for the other three datasets. Although theoretically, our method as well as all previous methods can predict object-object links containing any number of objects, we set this limit to prevent the number of training/test samples from exponentially growing.

The O-A task

For the training samples, we first enumerate all object-attribute links in C and label them as positive training samples. That is, we are to enumerate every object-attribute pair (u,v) such that $u \in U, v \in V$ and $(u,v) \in E$. Then, we are to randomly generate the same number of negative training samples as the positive training samples, with each negative training sample being an object-attribute link which does not exist in C. Formally, each negative training sample (u,v) should satisfy $u \in U, v \in V$ and $(u,v) \notin E$.

For the test samples, we first enumerate all object-attribute links in C' which does not exist in C. These links are all labeled as positive test samples. That is, we are to enumerate every pair (u,v) such that $u \in U, v \in V$ and $(u,v) \in E' - E$. Then, we are to randomly generate the same number of negative training samples as the positive training samples, with each negative training sample being an object-attribute link which does not exist in C'. Formally, each negative test sample (u,v) should satisfy $u \in U, v \in V$ and $(u,v) \notin E'$.

5.5.3 Evaluation metrics

To give a fair and comprehensive evaluation, we use the following three measures: F_1 *score*, *AUC score*, and *AUPR score*. All three scores are estimated with the four basic values: TP, TN, FP, and FN. TP represents the number of samples that are positive and are predicted positive. FN represents the number of samples that are negative but are falsely predicted to be positive. TN represents the number of samples that are negative but are predicted negative. FN represents the number of samples that are positive but falsely predicted negative.

The F_1 score is the harmonic mean of the *precision* and *recall*. Precision, recall, and F_1 are estimated as follows:

$$\begin{aligned} Percision &:= \frac{TP}{TP+FP}, \\ Recall &:= \frac{TP}{TP+FN}, \\ F_1 &:= \frac{2}{recall^{-1}+percision^{-1}}. \end{aligned} \tag{5.3}$$

The F_1 value may vary as we change the threshold for the prediction score. Hence, in this research, for each test case, we try 20 different thresholds with the following procedure – the initial threshold is set to 0, and at each trial, we add the threshold by 0.05. After all trials, we report the highest F_1 value we get.

The AUC (Area Under the Curve) score is estimated by computing the area under the ROC (Receiver Operating Characteristic) curve. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. TPR and FPR are estimated as follows:

$$\begin{aligned} TPR &:= \frac{TP}{TP+FN}, \\ FPR &:= \frac{FP}{TN+FP}. \end{aligned} \tag{5.4}$$

The AUPR (Area Under the Precision-Recall Curve) score is estimated by computing the area under the Precision-Recall curve, which is created by plotting the precision rate against the *recall* rate at various threshold settings.

5.5.4 Experimental environment and parameters

Experimental environment: The experiments are conducted on a Windows 11 server with 64GB RAM, an AMD Ryzen 9 7900X CPU, and an NVIDIA GeForce RTX 4090 GPU.

The codes for the deep-learning part of our method are implemented with Python 3.8.18 and Pytorch 2.1.0. The code for the algorithm for extracting formal concepts is written in C++ and complied by MinGW 13.2.0.

Parameters: The dimension of input embeddings of the Transformer encoder is 768, the dimension of hidden layer in Transformer encoder is 3072.

For the BERT pre-training step, the maximum number of masked tokens used in MTP is set to 4, the dimension of embeddings is set to 768, the number of heads is set to 12, the number of Transformer encoders is set to 6.

For the BERT fine-tuning step, the dimension of the hidden layer is set to 512. The batch size is set to 24.

5.5.5 The experiment of the O-O task

We evaluate the performance of BERT4FCA in the O-O task on the five datasets. The competitor methods used in this experiment include an FCA4SML method *object2vec* [28], a widely-used classic method *Node2Vec* [40], two widely-used classic hyperedge prediction methods, *common neighbors* [86] and *Katz index* [50], and a relatively new hyperedge prediction method *HPRA* [61]. Note that for the object2vec method, both CBoW and SG models are tested. For the Node2Vec method, after obtaining embedding vectors for each object and attribute from the input network, we train a downstream Logistic Regression classifier using the labeled training samples generated with the procedure introduced above. For each sample G, its feature is set to the concatenation of the embedding vectors of every object $g \in G$. The objects are sorted in the lexicographical order before concatenation. If a sample has fewer than ε_p objects, its feature vector is padded with zeros to make sure that all samples' feature vectors are of the same dimension. For the three hyperedge prediction methods, as introduced previously, they can be directly applied to the O-O task simply because the O-O task is a special case of hyperedge prediction.

The results are shown in Table 5.2.

The results show that BERT4FCA outperforms the other models across all datasets. We have also discovered that although Node2Vec has achieved high scores close to our method on the BMS-POS dataset, its performances on the other datasets are much lower than our method. This shows the stability of our method, *i.e.*, we can have a stable and high performance across different datasets. Also, as can be seen from the result, the three hyperedge prediction methods' performance are far better than object2vec, while our BERT4FCA can still outperform all of them. This implies that the ideology of utilizing the information in concept lattice for link prediction is valuable, while object2vec does not utilizes it very well. Our BERT4FCA, on the other hand, has fully learned and utilizes the information

from the concept lattice and thus has achieved good results. We will conduct extra ablation experiments later in this section to prove that our inference is correct.

5.5.6 The experiment of the O-A task

We evaluate the performance of BERT4FCA in the O-A task on all datasets. We compare BET4FCA with an FCA4RML method, *Structure Hole* [121], two widely-used non-FCA-based classic methods – *Node2Vec* and *matrix factorization with singular value decomposition* (MF-SVD), and a relatively new deep-learning method, *SBGNN* [45]. For the Node2Vec method, after obtaining embedding vectors for each object and attribute from the input network, we train a downstream Logistic Regression classifier using the labeled training samples generated with the procedure introduced above. For each sample (u, v), its feature is set to the concatenation of the embedding vectors of object u and attribute v.

The results are shown in Table 5.3.

From the results, we can see that among all five datasets, BERT4FCA has the best overall performance. Its performance is notably better than Structure Hole and generally better than SBGNN. For MF-SVD and Node2Vec, although they have achieved higher scores than BERT4FCA in some metrics in some dataset, their overall performance is mixed, while our BERT4FCA has an overall good performance acorss five datasets. The reason why these two methods get mixed performance is analyzed below.

In the BMS-POS dataset, MF-SVD exhibits higher AUC scores than other methods but with very low AUPR scores. A low AUPR score indicates that the model's high prediction scores do not correlate well with being in the positive class, suggesting that the model has difficulty achieving high precision. Since in link prediction, we focus on predicting the generation of new links, which is predicting the positive samples but not the negative samples, so if a model exhibits a low AUPR, it is considered to have low performance in link prediction. The low performance of MF-SVD may suggest that the underlying latent features of the networks and the relationships between nodes are intricate and challenging for MF to capture effectively. Additionally, since the BMS-POS dataset is extracted from a large dataset, maybe the relationships between nodes in this network are not consistent and stable, resulting in low prediction performance of MF.

In the iJO1366 dataset, Node2Vec exhibits higher AUPR and AUC scores than our method. However, its scores in the Review dataset is far lower than our method, indicating that the performance of Node2Vec varies greatly across different datasets. According to previous research [20], the sparser the network is, the better the process can capture the features of the network. Hence, it is expected to work better on sparse networks. Since iJO1366 is the most sparse data set, Node2Vec achieved to gain a better performance, but for

relative dense networks, its performances become notably lower. Hence, we consider our method to be more practical than Node2Vec thanks to its overall good performance in all metrics across all datasets.

Finally, we have also found that the performance of the other FCA4RML method, Structure Hole, is notably lower on these five datasets than the datasets used in [121]. As analyzed before, Structure Hole is a rule-based method that uses the same simple rule to extract information from the concept lattices and make predictions on all datasets, which will be highly likely to have lower performance on some datasets because not all datasets fit the rule well, and our experimental results just proved it. This again shows the importance of developing a method like our BERT4FCA that can automatically capture the information from concept lattices using statistical machine-learning techniques.

Table 5.2 The results for the O-O tasks. Note that O2V stands for object2vec. CN stands for common neighbors.

Method	ICFCA			BMS-POS			Keyword-Paper			Review			iJO1366		
	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR
O2V-CBoW	0.686	0.691	0.672	0.690	0.637	0.560	0.463	0.504	0.518	0.614	0.543	0.549	0.579	0.581	0.527
O2V-SG	0.652	0.497	0.519	0.676	0.649	0.577	0.236	0.290	0.388	0.527	0.501	0.480	0.612	0.674	0.675
Node2Vec	0.587	0.703	0.751	0.870	0.946	0.945	0.438	0.647	0.635	0.667	0.611	0.613	0.671	0.630	0.621
HPRA	0.752	0.808	0.767	0.862	0.927	0.824	0.722	0.761	0.675	0.675	0.724	0.668	0.956	0.986	0.889
CN	0.741	0.794	0.777	0.872	0.929	0.824	0.729	0.784	0.701	0.667	0.709	0.684	0.962	0.986	0.889
Katz	0.752	0.779	0.754	0.853	0.918	0.814	0.724	0.773	0.688	0.686	0.726	0.681	0.819	0.748	0.757
BERT4FCA	**0.781**	**0.877**	**0.896**	**0.871**	**0.964**	**0.963**	**0.800**	**0.911**	**0.908**	**0.765**	**0.788**	**0.758**	**0.965**	**0.994**	**0.994**

Table 5.3 The results for the O-A tasks.

Method	ICFCA			BMS-POS			Keyword-Paper			Review			iJO1366		
	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR
Structure Hole	0.018	0.000	0.000	0.226	0.000	0.000	0.005	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Node2Vec	0.671	0.800	0.764	0.612	0.793	0.755	0.661	0.747	0.697	0.630	0.579	0.566	0.690	**0.757**	**0.820**
MF-SVD	**0.749**	0.798	0.508	**0.824**	**0.892**	0.641	0.733	0.693	0.424	0.531	0.487	0.069	0.630	0.730	0.549
SBGNN	0.701	0.710	0.652	0.816	0.803	0.731	0.658	0.655	0.615	0.623	0.556	0.530	0.602	0.700	0.679
BERT4FCA	0.741	**0.812**	**0.777**	0.725	0.823	**0.788**	**0.744**	**0.765**	**0.711**	**0.677**	**0.675**	**0.690**	**0.697**	0.741	0.757

5.5.7 Ablation experiments

To verify if our original mechanisms are functioning well, we conduct two ablation experiments – the first one checks if our method can indeed learn the neighboring relations of formal concepts; the second one checks if the information in concept lattices our method learned is indeed helpful for making link predictions.

In the first ablation experiment, we only use 80% of the training samples to pre-train both the object model and the attribute model on all datasets. The remaining 20% of the original training samples are kept as test samples for the NCP task – these intents/extents pairs are not presented in the training samples, so if our method can correctly predict whether they are neighbors or not, it should indicate that our method has learned the structure of concept lattice well. The results are shown in Table 5.4.

Table 5.4 The results for the first supplementary experiment.

Dataset	Object/Attribute	F_1	AUC	AUPR
ICFCA	Object	0.903	0.896	0.842
	Attribute	0.868	0.842	0.785
BMS-POS	Object	0.993	0.993	0.987
	Attribute	0.978	0.978	0.963
Keyword-Paper	Object	0.965	0.966	0.934
	Attribute	0.900	0.902	0.850
Review	Object	0.938	0.826	0.882
	Attribute	0.857	0.706	0.786
iJO1366	Object	0.977	0.990	0.956
	Attribute	0.880	0.918	0.796

The results suggest that BERT4FCA indeed learned the neighboring relations from the concept lattice on all datasets well. We have also noticed that the results of the object models are generally better than the attribute models across all datasets. We analyze it because, in these datasets, the average lengths of intents are longer than that of the extents, making it potentially more challenging to effectively capture the neighboring relations of intents.

In the second ablation experiment, we check if the information we learned from concept lattices indeed contributes to better link prediction results. Since the information of the concept lattices is learned in the NCP and MTM tasks in the pre-training step, in this experiment, we skip the pre-training step and directly fine-tune the models with randomly initialized weights. By comparing the results with that of our full method, we will know whether our specially designed tasks for learning the information of concept lattices are functioning well. The results for the O-O tasks are shown in Table 5.5, and the results for the O-A tasks are shown in Table 5.6.

Table 5.5 The results for the O-O tasks of the second ablation experiment.

Method	ICFCA			BMS-POS			Keyword-Paper			Review			iJO1366		
	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR
BERT4FCA-NC	0.718	0.810	0.815	0.865	0.961	0.958	0.735	0.866	0.858	0.719	0.730	0.719	0.940	0.985	0.987
BERT4FCA	**0.781**	**0.877**	**0.896**	**0.871**	**0.964**	**0.963**	**0.800**	**0.911**	**0.908**	**0.765**	**0.788**	**0.758**	**0.965**	**0.994**	**0.994**
Improvement	0.063	0.067	0.081	0.006	0.003	0.005	0.065	0.045	0.050	0.046	0.058	0.039	0.025	0.009	0.007

Table 5.6 The results for the O-A tasks of the second ablation experiment.

Method	ICFCA			BMS-POS			Keyword-Paper			Review			iJO1366		
	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR	F_1	AUC	AUPR
BERT4FCA-NC	0.723	0.801	0.766	0.701	0.800	0.760	0.724	0.748	0.707	0.667	0.660	0.629	0.695	0.740	0.749
BERT4FCA	**0.741**	**0.812**	**0.777**	**0.725**	**0.823**	**0.788**	**0.744**	**0.765**	**0.711**	**0.677**	**0.675**	**0.690**	**0.697**	**0.741**	**0.757**
Improvement	0.018	0.011	0.011	0.024	0.023	0.028	0.020	0.017	0.004	0.010	0.015	0.061	0.002	0.001	0.008

The results show that learning more information from concept lattices indeed contributes to the improvement of performances in both tasks, while the degree of improvement varies across different datasets and tasks. The overall degree of improvement in O-O tasks is higher than that in the O-A task. We analyze it because compared to the attribute models, the object models better capture the information on concept lattices, such as the neighboring relations, as demonstrated in the first ablation experiment. The difference in improvements shows that better learning the neighboring relations contributes to the prediction performance.

Also, we find that in the O-O task on the BMS-POS dataset, learning the information from concept lattices contributes to the least degree of improvement. This may be because the performance of our method without learning the information is already sufficiently high, as the AUC and AUPR scores have reached 96%. In such a case, it is considered hard to improve the performance further, even with an effective mechanism.

Furthermore, we observe that even without the pre-training step, our method can still have a relatively good performance. This indicates the BERT framework is capable of extracting information to make predictions even directly from the raw bipartite network. This discovery may provide hints for further research on the possibility of training a light-weight model with a small part of the network utilizing the effective BERT-like transformer network structure. Nevertheless, when there is an urge requirement of stability and quality, it is recommended to apply our full BERT4FCA model as the information from maximal bi-cliques extracted by our proposed MTP and NCP tasks can indeed further improve the performance.

5.6 Conclusion of the chapter

In this chapter, we proposed BERT4FCA, a novel FCA4SML method for learning the information in the concept lattices and use them for downstream machine learning tasks. The experimental results demonstrated that our methods outperform previous FCM4SML methods like object2Vec and non-FCA-based classic methods like MF and Node2Vec in both the O-O and the O-A tasks on five different datasets. The results have shown that our method is stabler than previous methods. We also demonstrated by ablation experiments that neighboring relations between maximal bi-cliques are well learned by the model as expected and that such information contributes to better link prediction results. Furthermore, we have shown that BERT4FCA provides a general framework for employing BERT to learn the information extracted by FCA. Hence, in addition to these tasks, we believe BERT4FCA can also be further applied to a broader range of real-world tasks. We plan to explore the potential applications in the future and make BERT4FCA become the first widely applicable FCA4SML framework.

However, we have also found some points of our method that may be further refined and improved. First, limited by computational resources, we were unable to pre-train an all-in-one model for both objects and attributes in this research. If such a model can be trained well, it will be more convenient as all downstream tasks will only need one single pre-trained model. Second, although BERT4FCA outperforms previous methods in prediction results, the pre-training step is time-consuming, especially when dealing with datasets with a large number of formal concepts. With our current computational resources, to finish the pre-training step in a reasonable time, the maximum number of concepts should be around 15,000. Although there have been methods for reducing the number of formal concepts in concept lattices, it is uncertain whether training on a reduced concept lattice will result in lower performances. If so, we may need to work out a technique to make a trade-off between the training time and the performance. We plan to address these two points in our future work.

Chapter 6

Conclusion

6.1 Summary of our contributions

In this thesis, we have proposed four methods contributing to the integration of FCA and SML.

- In Chapter 2, we have proposed the algorithm *Z-TCA*, an algorithm utilizing ZDDs to speed up the process of FCA and TCA. The algorithm is especially efficient in denser contexts and is about three times as fast as previous methods.
- In Chapter 3, we have proposed a method for reducing the number of objects and concept lattices using integer linear programming. The method is more reliable than previous method because it has the ability to prevent the context and the concept lattice from being modified too much.
- In Chapter 4, we have proposed an FCA4SML method for link prediction in bipartite networks. This method avoids direct processing of the concept lattice with the SML method, but instead uses a rule-based method to process the concept lattice and generate training samples for the SML method. Our experimental results have shown that the process contributes to good performance.
- In Chapter 5, we have proposed a generally applicable FCA4SML method called *BERT4FCA*. The method utilizes a BERT-like Transformer encoder network to efficiently capture both types of information in the concept lattice. The method shows its great performance in two machine learning tasks – link prediction and hyper-link prediction on bipartite networks. Our ablation experiments have also shown that the information in the concept lattice is indeed captured in the trained model, and the captured information indeed contribute to the good performance.

Among the four methods we have proposed, two of these are general improvements to FCA, so they can be also applied to any task that utilizes FCA like data mining or knowlege processing. The other two of these are not only successful integration of FCA and SML, but are also counted as the best-performing methods in the related machine learning tasks, that is, link prediction and hyper-link prediction in bipartite networks. Hence, we believe that our methods can not only contributes to further research on FCA4SML, but also open up new possibilities to the research on FCA itself and all related machine learning tasks.

6.2 Outlook on the future of FCA4SML

We now present our views on the possible future research trends related to the integration of FCA and SML. We consider that in the near future, the main purpose of research on FCA4SML will still focus on how to use FCA to help the preceding SML process capture more information from the data. However, ever since the introduction of *large language models* (LLM) [122], SML methods' capability of capturing information has increased sharply – the GPT3.5 and GPT4 models used for the well-known *ChatGPT* service [1] has already shown a strong ability to capture relatively deep latent information within the most unprocessed data like raw text [1]. Hence, we predict that advanced SML methods will eventually become capable of directly retrieving information and features from any source of raw data. At that time, we predict that the main purpose of integrating FCA with the SML methods will shift to **preventing the hallucination** and **increasing the reliability** of advanced SML methods utilizing large generative models. The way of integrating FCA with SML will also become "reversed" – instead of using SML methods to capture the information in concept lattices, future research at that time will focus on using SML methods to decode the captured information in large generative models back into concept lattices.

Currently, although large generative models are believed to already have the ability to retrieve information from raw data, methods utilizing these models suffer from the problem of *hallucination* [49], that is, these methods may frequently generate outputs that appear factual but are actually unfaithful to the input reference data. It has been conjectured that a possible cause of hallucination is that the information from the raw data is only retrieved and stored in the form of weight matrices in the generative model but not correctly organized into knowledge and thus the model cannot precept and process it before generating outputs. To verify this conjecture, we need to analyze how the information from the raw data is stored in the model and how the model utilizes the information to generate outputs, which is impossible because, inside the model, the information is encoded into large numbers of parameters which is not understandable for human beings [122]. Nevertheless, inspired by

the idea of knowledge representation that aims to represent knowledge in a structural form that is both comprehensible for human beings and easy for machine processing, we have come up with the idea to use another SML process to simulate the knowledge extraction and representation process. That is, we aim to train a decoding model to decode the information stored within the generative model back into a structural knowledge representation form that can be understood by human beings. With such a decoding model, we can understand what knowledge has been learned by the model and from what knowledge did they generate the output, which greatly improves the reliability of the generative model.

Recently, methods for generating *knowledge graphs* from LLMs have already been proposed for relieving the hallucination and increasing the reliability of generative models [89, 117]. Since for some specific types of data like biological taxonomy, concept lattices are better suitable as the form of knowledge representation thanks to their capability of showing the conceptual hierarchy, we believe that it is also of high value to develop such an SML method that simulates the FCA process to decode the information in the generative model back into concept lattices. Compared to the traditional way of using SML method to capture the information from the concept lattice, such a "reversed" integration of FCA and SML is far more difficult because to the furthest of our knowledge, it is still unclear whether the strict conceptual hierarchical structure of concept lattices can be generated by SML methods. Nevertheless, if such a method can be worked out, it can not only help in reducing the hallucination problem of generative models, but also help us build up closer connection between FCA and SML and thus open up new possibilities to both fields.

Although in this thesis, we did not directly study on the reversed integration of FCA and SML, some of our contributions may also be used in future research on such a topic. For example, in Chapter 2, we have proposed formal methods for checking the reliability of an object reduction process. These methods may also be used for working out or evaluating an SML method that simulates the object reduction process. Also, in Chapter 5, our method for using the two tasks in BERT to capture the information in concept lattices may also provide hints on designing the structure of network for training the decoding model. Hence, we believe our contributions will have a long-term influence in related fields.

6.3 Future work

The future work for each sub-topic related to our proposed methods has already been discussed at the end of each chapter. Hence, in this section, we are to list some planned work related to the general topic of the thesis, that is, the integration of FCA and SML. As introduced in the last section, in the near future, the studies of integration of FCA and

SML will still focus on using FCA to boost the performance of preceding SML methods; in relatively further future, the studies on this topic will shift to the reversed version, that is, to use the SML methods to decode the information learned by the generative models back into the concept lattices. Hence, our plan for future work will also follow our prediction – we plan to first continue research on the traditional integration of FCA and SML before shifting to the more challenging reversed integration.

Firstly, in this thesis, we only studied the application of the FCA4SML method on machine learning tasks that take binary relational data as input. However, over the years, real-world data have becoming more and more complex and may be hard to be represented into such a clean form of binary relational tables. In order to broaden the scope of applications of FCA4SML, we plan to develop an FCA4SML method that can take more complex forms of data as input, probably with the help of pre-processing methods or variations of FCA (including the natural extension TCA).

Secondly, previous FCA4SML methods, including the two methods proposed in this thesis, are all *supervised learning* methods. However, there have been successful *unsupervised* FCA4RML methods in tasks like node clustering, showing the potential possibility of developing an unsupervised FCA4SML method [93]. Hence, we also plan to develop an unsupervised FCA4SML method, which may further broaden the scope of applications of FCA4SML.

Thirdly, as a long-term plan, we plan to develop the aforementioned "reversed" integration of FCA and SML. For the first step, we plan to work out a simple version of the integration – that is, to use an SML process to simulate a basic FCA process that constructs the concept lattice from formal contexts. A related topic has been discussed in [28] where the authors proposed a method to use neural networks to simulate the derivation operators within a context. The method is proven to work but only has a limited performance. Hence, we plan to follow up their research and improve their methods first before trying to work out completely new methods. After we achieve such a simple version of reversed integration, we then plan to challenge the full version for decoding the learned information in the large generative models back into concept lattices.

References

[1] Abdullah, M., Madain, A., and Jararweh, Y. (2022). Chatgpt: Fundamentals, applications and social impacts. In *2022 Ninth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, pages 1–8. Ieee.

[2] Akers, S. B. (1978). Binary decision diagrams. *IEEE Transactions on computers*, 27(06):509–516.

[3] Andersen, M., Dahl, J., and Vandenberghe, L. (2020). CVXOPT: Convex optimization. Astrophysics Source Code Library, record ascl:2008.017.

[4] Asratian, A. S., Denley, T. M., and Häggkvist, R. (1998). *Bipartite graphs and their applications*, volume 131. Cambridge University Press.

[5] Balogh, O. M., Benczik, B., Horváth, A., Pétervári, M., Csermely, P., Ferdinandy, P., and Ágg, B. (2022). Efficient link prediction in the protein–protein interaction network using topological information in a generative adversarial network machine learning model. *BMC bioinformatics*, 23(1):1–19.

[6] Baptista, A., Gonzalez, A., and Baudot, A. (2022). Universal multilayer network exploration by random walk with restart. *Communications Physics*, 5(1):170.

[7] Barabási, A. and Albert, R. (1999). Emergence of scaling in random networks. *science*, 286(5439):509–512.

[8] Bazin, A., Galasso, J., and Kahn, G. (2024). Polyadic relational concept analysis. *International Journal of Approximate Reasoning*, 164:109067.

[9] Belohlavek, R., De Baets, B., Outrata, J., and Vychodil, V. (2009). Inducing decision trees via concept lattices. *International journal of general systems*, 38(4):455–467.

[10] Bodon, F. (2003). A fast apriori implementation. In *Proceedings of FIMI'03 Workshop on Frequent Itemset Mining Implementations*, pages 16–25.

[11] Brown, W. J., Malveau, R. C., McCormick III, H. W., and Mowbray, T. J. (1998). *AntiPatterns Refactoring Software, Architectures, and Projects in Crisis*. Robert Ipsen.

[12] Cantador, I., Brusilovsky, P., and Kuflik, T. (2011). 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA. ACM.

[13] Chen, B., Li, F., Chen, S., Hu, R., and Chen, L. (2017). Link prediction based on non-negative matrix factorization. *PLOS ONE*, 12(8):e0182968.

[14] Chen, F., Zhang, Z., Li, J., Kang, J., and Yang, H. (2009). Service identification via ontology mapping. In *2009 33rd Annual IEEE International Computer Software and Applications Conference*, volume 1, pages 486–491. IEEE.

[15] Chen, X., Jiao, P., Yu, Y., Li, X., and Tang, M. (2019). Toward link predictability of bipartite networks based on structural enhancement and structural perturbation. *Physica A: Statistical Mechanics and its Applications*, 527:121072.

[16] Chen, X., Wang, W., Yu, W., Jiao, P., and Sun, Y. (2018). Highlighting link prediction in bipartite networks via structural perturbation. *IEEE Access*, 6:73583–73592.

[17] Cheung, K. S. and Vogel, D. (2005). Complexity reduction in lattice-based information retrieval. *Information retrieval*, 8(2):285–299.

[18] Chu, X., Sun, B., Chu, X., Wu, J., Han, K., Zhang, Y., and Huang, Q. (2022). Multi-granularity dominance rough concept attribute reduction over hybrid information systems and its application in clinical decision-making. *Information Sciences*, 597:274–299.

[19] Conforti, M., Cornuéjols, G., Zambelli, G., et al. (2014). *Integer programming*, volume 271. Springer, Heidelberg.

[20] Dalmia, A. and Gupta, M. (2018). Towards interpretation of node embeddings. In *Companion Proceedings of the The Web Conference 2018*, pages 945–952.

[21] Davey, B. and Priestley, H. (2002). *Introduction to Lattices and Order*. Cambridge University Press, New York, 2 edition.

[22] Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240.

[23] Derr, T., Johnson, C., Chang, Y., and Tang, J. (2019). Balance in signed bipartite networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1221–1230.

[24] Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[25] Dias, S. M. and Vieira, N. (2010). Reducing the size of concept lattices: The jbos approach. In *Cla*, volume 672, pages 80–91.

[26] Dias, S. M. and Vieira, N. J. (2015). Concept lattices reduction: Definition, analysis and classification. *Expert Systems with Applications*, 42(20):7084–7097.

[27] Dias, S. M. and Vieira, N. J. (2017). A methodology for analysis of concept lattice reduction. *Information Sciences*, 396:202–217.

[28] Dürrschnabel, D., Hanika, T., and Stubbemann, M. (2021). Fca2vec: Embedding techniques for formal concept analysis. In *Complex Data Analytics with Formal Concept Analysis*, pages 47–74. Springer, Cham.

[29] Estmark, A. (2021). Text block prediction and article reconstruction using bert.

[30] Falleri, J., Huchard, M., and Nebut, C. (2008). A generic approach for class model normalization. In *2008 23rd IEEE/ACM International Conference on Automated Software Engineering*, pages 431–434. IEEE.

[31] Ferré, S. and King, R. D. (2004). Blid: an application of logical information systems to bioinformatics. In *International Conference on Formal Concept Analysis*, pages 47–54. Springer.

[32] Fokoue, A., Sadoghi, M., Hassanzadeh, O., and Zhang, P. (2016). Predicting drug-drug interactions through large-scale similarity-based link prediction. In *The Semantic Web. Latest Advances and New Domains: 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29–June 2, 2016, Proceedings 13*, pages 774–789. Springer.

[33] Forrest, J., Ralphs, T., Santos, H. G., Vigerske, S., Hafer, L., Forrest, J., Kristjansson, B., jpfasano, EdwinStraver, Lubin, M., rlougee, jpgoncal1, Jan-Willem, h-i gassmann, Brito, S., Cristina, Saltzman, M., tosttost, MATSUSHIMA, F., and to st (2022). coin-or/cbc: Release releases/2.10.7.

[34] Fu, H., Fu, H., Njiwoua, P., and Nguifo, E. M. (2004). A comparative study of fca-based supervised classification algorithms. In *International Conference on Formal Concept Analysis (ICFCA) 2004*, pages 313–320. Springer.

[35] Ganter, B. (2010). Two basic algorithms in concept analysis. In *International conference on formal concept analysis*, pages 312–340. Springer.

[36] Ganter, B. and Kuznetsov, S. O. (2000). Formalizing hypotheses with concepts. In *International Conference on Conceptual Structures (ICCS) 2000*, pages 342–356. Springer.

[37] Ganter, B. and Wille, R. (1999). *Formal concept analysis: mathematical foundations*. Springer-Verlag, Berlin Heidelberg, 1 edition.

[38] Getoor, L. and Diehl, C. P. (2005). Link mining: a survey. *Acm Sigkdd Explorations Newsletter*, 7(2):3–12.

[39] Graves, A. and Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610.

[40] Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864.

[41] Gurobi Optimization, LLC (2023). Gurobi Optimizer Reference Manual.

[42] Han, J., Pei, J., and Yin, Y. (2000). Mining frequent patterns without candidate generation. *ACM sigmod record*, 29(2):1–12.

[43] Harper, F. M. and Konstan, J. A. (2015). The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19.

[44] Hassan, B. A., Rashid, T. A., and Mirjalili, S. (2021). Formal context reduction in deriving concept hierarchies from corpora using adaptive evolutionary clustering algorithm star. *Complex & Intelligent Systems*, 7(5):2383–2398.

[45] Huang, J., Shen, H., Cao, Q., Tao, S., and Cheng, X. (2021). Signed bipartite graph neural networks. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 740–749.

[46] Ignatov, D. I. (2015). Introduction to formal concept analysis and its applications in information retrieval and related fields. *Information Retrieval: 8th Russian Summer School, RuSSIR 2014, Nizhniy Novgorod, Russia, August 18-22, 2014, Revised Selected Papers 8*, pages 42–141.

[47] Ishigure, H., Mutoh, A., Matsui, T., and Inuzuka, N. (2015). Concept lattice reduction using attribute inference. In *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)*, pages 108–111. IEEE.

[48] Jaschke, R., Hotho, A., Schmitz, C., Ganter, B., and Stumme, G. (2006). Trias–an algorithm for mining iceberg tri-lattices. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 907–911. IEEE.

[49] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

[50] Katz, L. (1953). A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43.

[51] Kaya, B. (2020). Hotel recommendation system by bipartite networks and link prediction. *Journal of Information Science*, 46(1):53–63.

[52] King, Z. A., Lu, J., Dräger, A., Miller, P., Federowicz, S., Lerman, J. A., Ebrahim, A., Palsson, B. O., and Lewis, N. E. (2015). BiGG Models: A platform for integrating, standardizing and sharing genome-scale models. *Nucleic Acids Research*, 44(D1):D515–D522.

[53] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

[54] Kipf, T. N. and Welling, M. (2016). Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*.

[55] Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

[56] Kosub, S. (2019). A note on the triangle inequality for the jaccard distance. *Pattern Recognition Letters*, 120:36–38.

[57] Krolak-Schwerdt, S., Orlik, P., and Ganter, B. (1994). Tripat: a model for analyzing three-mode binary data. In *Information systems and data analysis*, pages 298–307. Springer.

[58] Krupka, M. (2012). On complexity reduction of concept lattices: three counterexamples. *Information retrieval*, 15(2):151–156.

[59] Kumar, C. A. and Srinivas, S. (2010). Concept lattice reduction using fuzzy k-means clustering. *Expert systems with applications*, 37(3):2696–2704.

[60] Kumar, P. and Sharma, D. (2020). A potential energy and mutual information based link prediction approach for bipartite networks. *Scientific Reports*, 10(1):1–14.

[61] Kumar, T., Darwin, K., Parthasarathy, S., and Ravindran, B. (2020). Hpra: Hyperedge prediction using resource allocation. In *Proceedings of the 12th ACM conference on web science*, pages 135–143.

[62] Kunegis, J., De Luca, E. W., and Albayrak, S. (2010). The link prediction problem in bipartite networks. In *Computational Intelligence for Knowledge-Based Systems Design: 13th International Conference on Information Processing and Management of Uncertainty, IPMU 2010, Dortmund, Germany, June 28-July 2, 2010. Proceedings 13*, pages 380–389. Springer.

[63] Lazega, E. (1995). Structural holes: the social structure of competition.

[64] Lehmann, F. and Wille, R. (1995). A triadic approach to formal concept analysis. In *International conference on conceptual structures*, pages 32–43. Springer.

[65] Li, J., Kumar, C. A., Mei, C., and Wang, X. (2017). Comparison of reduction in formal decision contexts. *International Journal of Approximate Reasoning*, 80:100–122.

[66] Li, J., Mei, C., and Lv, Y. (2013). Incomplete decision contexts: approximate concept construction, rule acquisition and knowledge reduction. *International Journal of Approximate Reasoning*, 54(1):149–165.

[67] Maddouri, M. (2005). A formal concept analysis approach to discover association rules from data. In *Proceedings of the CLA 2005 International Workshop on Concept Lattices and their Applications*, pages 10–21. Bělohlávek R., Snášel V. (Eds.).

[68] Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, 9(1):381–386.

[69] Marquer, E., Kulkarni, A., and Couceiro, M. (2020). Embedding formal contexts using unordered composition. In *FCA4AI-8th International Workshop "What can FCA do for Artificial Intelligence?" (co-located with ECAI2020)*.

[70] Martínez, V., Berzal, F., and Cubero, J. (2016). A survey of link prediction in complex networks. *ACM computing surveys (CSUR)*, 49(4):1–33.

[71] Martínez, V., Berzal, F., and Cubero, J. C. (2016). A survey of link prediction in complex networks. *ACM Computing Surveys (CSUR)*, 49:1 – 33.

[72] Maurya, D. and Ravindran, B. (2021). Hyperedge prediction using tensor eigenvalue decomposition. *Journal of the Indian Institute of Science*, 101:443–453.

[73] Menon, A. K. and Elkan, C. (2011). Link prediction via matrix factorization. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011, Proceedings, Part II 22*, pages 437–452. Springer.

[74] Mi, J., Leung, Y., and Wu, W. (2010). Approaches to attribute reduction in concept lattices induced by axialities. *Knowledge-Based Systems*, 23(6):504–511.

[75] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[76] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

[77] Minato, S. (2001). Zero-suppressed bdds and their applications. *International Journal on Software Tools for Technology Transfer*, 3(2):156–170.

[78] Minato, S. and Arimura, H. (2007). Frequent closed item set mining based on zero-suppressed bdds. *Information and Media Technologies*, 2(1):309–316.

[79] Minato, S., Uno, T., and Arimura, H. (2008). Lcm over zbdds: Fast generation of very large-scale frequent itemsets using a compact graph-based representation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 234–246. Springer.

[80] Moha, N., Rouane Hacene, A. M., Valtchev, P., and Guéhéneuc, Y. (2008). Refactorings of design defects using relational concept analysis. In *Formal Concept Analysis: 6th International Conference, ICFCA 2008, Montreal, Canada, February 25-28, 2008. Proceedings 6*, pages 289–304. Springer.

[81] Mongia, A., Sengupta, D., and Majumdar, A. (2019). Mcimpute: matrix completion based imputation for single cell rna-seq data. *Frontiers in genetics*, 10:9.

[82] Nasiri, E., Berahmand, K., Rostami, M., and Dabiri, M. (2021). A novel link prediction algorithm for protein-protein interaction networks by attributed graph embedding. *Computers in Biology and Medicine*, 137:104772.

[83] Nassar, H., Benson, A. R., and Gleich, D. F. (2020). Neighborhood and pagerank methods for pairwise link prediction. *Social Network Analysis and Mining*, 10:1–13.

[84] Neto, S. M., Zárate, L. E., and Song, M. A. (2018). Handling high dimensionality contexts in formal concept analysis via binary decision diagrams. *Information Sciences*, 429:361–376.

[85] Neves, J., Ananias, K., Nobre, C., Zarate, L., and Song, M. (2020). Applying binary decision diagram to extract concepts from triadic formal context. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pages 466–469.

[86] Newman, M. E. (2001). Clustering and preferential attachment in growing networks. *Physical review E*, 64(2):025102.

[87] Nickel, M., Murphy, K., Tresp, V., and Gabrilovich, E. (2015). A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33.

[88] Ou-Yang, L., Lu, F., Zhang, Z., and Wu, M. (2022). Matrix factorization for biomedical link prediction and scrna-seq data imputation: an empirical survey. *Briefings in Bioinformatics*, 23(1):bbab479.

[89] Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., and Wu, X. (2024). Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*.

[90] Pasquier, N., Bastide, Y., Taouil, R., and Lakhal, L. (1999). Discovering frequent closed itemsets for association rules. In *International Conference on Database Theory*, pages 398–416. Springer.

[91] Pei, D. and Mi, J. (2011). Attribute reduction in decision formal context based on homomorphism. *International Journal of Machine Learning and Cybernetics*, 2:289–293.

[92] Poelmans, J., Ignatov, D. I., Kuznetsov, S. O., and Dedene, G. (2013a). Formal concept analysis in knowledge processing: A survey on applications. *Expert systems with applications*, 40(16):6538–6560.

[93] Poelmans, J., Kuznetsov, S. O., Ignatov, D. I., and Dedene, G. (2013b). Formal concept analysis in knowledge processing: A survey on models and techniques. *Expert systems with applications*, 40(16):6601–6623.

[94] Rimsa, A., Song, M. A., and Zárate, L. E. (2013). Scgaz-a synthetic formal context generator with density control for test and evaluation of fca algorithms. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pages 3464–3470. IEEE.

[95] Roscoe, S., Khatri, M., Voshall, A., Batra, S., Kaur, S., and Deogun, J. (2022). Formal concept analysis applications in bioinformatics. *ACM Computing Surveys*, 55(8):1–40.

[96] Salha, G., Hennequin, R., and Vazirgiannis, M. (2019). Keep it simple: Graph autoencoders without graph convolutional networks. *arXiv preprint arXiv:1910.00942*.

[97] Shang, M., Lü, L., Zhang, Y., and Zhou, T. (2010). Empirical analysis of web-based user-object bipartite networks. *Europhysics Letters*, 90(4):48006.

[98] Shang, W., Sohn, K., Almeida, D., and Lee, H. (2016). Understanding and improving convolutional neural networks via concatenated rectified linear units. In *international conference on machine learning*, pages 2217–2225. PMLR.

[99] Shin, J., Gim, M., Park, D., Kim, S., and Kang, J. (2020). Bipartite link prediction by intra-class connection based triadic closure. *IEEE Access*, 8:140194–140204.

[100] Singh, P. K. and Kumar, C. A. (2017). Concept lattice reduction using different subset of attributes as information granules. *Granular computing*, 2(3):159–173.

[101] Snasel, V., Abdulla, H. M. D., and Polovincak, M. (2007a). Behavior of the concept lattice reduction to visualizing data after using matrix decompositions. In *2007 Innovations in Information Technologies (IIT)*, pages 392–396. IEEE.

[102] Snasel, V., Gajdos, P., Abdulla, H. M. D., and Polovincak, M. (2007b). Concept lattice reduction by matrix decompositions. In *The 1st International Conference on Digital Communications and Computer Applications (DCCA 2007)*, pages 35–40.

[103] Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., and Lakhal, L. (2002). Computing iceberg concept lattices with titanic. *Data & knowledge engineering*, 42(2):189–222.

[104] Tan, A., Li, J., and Lin, G. (2015). Connections between covering-based rough sets and concept lattices. *International journal of approximate reasoning*, 56:43–58.

[105] Uno, T., Asai, T., Uchida, Y., and Arimura, H. (2003). Lcm: An efficient algorithm for enumerating frequent closed item sets. In *Fimi*, volume 90.

[106] Vanschoren, J., van Rijn, J. N., Bischl, B., and Torgo, L. (2013). Openml: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60.

[107] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

[108] Voutsadakis, G. (2002). Polyadic concept analysis. *Order*, 19:295–304.

[109] Wang, P., Xu, B., Wu, Y., and Zhou, X. (2014). Link prediction in social networks: the state-of-the-art. *Science China Information Sciences*, 58:1–38.

[110] Wang, W., Chen, X., Jiao, P., and Jin, D. (2017). Similarity-based regularized latent feature model for link prediction in bipartite networks. *Scientific reports*, 7(1):16996.

[111] Wang, Z., Gerstein, M., and Snyder, M. (2009). Rna-seq: a revolutionary tool for transcriptomics. *Nature reviews genetics*, 10(1):57–63.

[112] Wille, R. (1997). Conceptual graphs and formal concept analysis. In *International Conference on Conceptual Structures*, pages 290–303. Springer.

[113] Wille, R. (2002). Why can concept lattices support knowledge discovery in databases? *Journal of Experimental & Theoretical Artificial Intelligence*, 14(2-3):81–92.

[114] Wu, T., Yin, M., Jiao, C., Gao, Y., Kong, X., and Liu, J. (2020). Mccmf: collaborative matrix factorization based on matrix completion for predicting mirna-disease associations. *BMC bioinformatics*, 21:1–22.

[115] Xia, S., Dai, B., Lim, E., Zhang, Y., and Xing, C. (2012). Link prediction for bipartite social networks: The role of structural holes. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 153–157. IEEE.

[116] Yadati, N., Nitin, V., Nimishakavi, M., Yadav, P., Louis, A., and Talukdar, P. (2020). Nhp: Neural hypergraph link prediction. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1705–1714.

[117] Yang, L., Chen, H., Li, Z., Ding, X., and Wu, X. (2023). Chatgpt is not enough: Enhancing large language models with knowledge graphs for fact-aware language modeling. *arXiv preprint arXiv:2306.11489*.

[118] Yang, S., Lu, Y., Jia, X., and Li, W. (2020). Constructing three-way concept lattice based on the composite of classical lattices. *International Journal of Approximate Reasoning*, 121:174–186.

[119] Zhang, A., Lipton, Z. C., Li, M., and Smola, A. J. (2023). *Dive into deep learning*. Cambridge University Press.

[120] Zhang, M. and Chen, Y. (2018). Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31.

[121] Zhao, J., Sun, M., Chen, F., and Chiu, P. (2019). Missbin: Visual analysis of missing links in bipartite networks. In *2019 IEEE Visualization Conference (VIS)*, pages 71–75. IEEE.

[122] Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., et al. (2023). A survey of large language models. *arXiv preprint arXiv:2303.18223*.

[123] Zheng, X., Ding, H., Mamitsuka, H., and Zhu, S. (2013). Collaborative matrix factorization with multiple similarities for predicting drug-target interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1025–1033.

Appendix A

A Running Example for our method in Chapter 3

We hereby introduce two simple running examples to better show the working flow of our method. In the first example, we are to reduce a sample context with our method applying the OR rule, and in the second example, we are to reduce another sample context applying the AND rule.

A.1 Example 1: ILP-based concept lattice reduction applying the OR rule

In this example, we are to use integer programming to reduce the formal context shown in the left panel of Fig. A.1. Its corresponding concept lattice is shown in the right panel of the figure.

Parameters and Variables

First, we are to determine the following hyper-parameters:

- ε_r, which represents the maximum allowed "distances" of two objects to be merged;
- ε_m, which represents the maximum allowed modifications to the context;
- ε_I, which represents the maximum allowed number of inserted concepts.

In this example, we set ε_r to 2, ε_m to 2, and ε_I to 0.

With the values of parameters determined, we are to derive matrix $\mathbf{C} = \{c_{ij}\}$, where c_{ij} is a Boolean variable representing whether the incidence x_{ij} is allowed to be modified. As

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
g_1	×				×	×	×			
g_2	×	×	×							
g_3	×	×		×						
g_4	×	×	×		×	×	×			
g_5	×	×		×				×	×	×

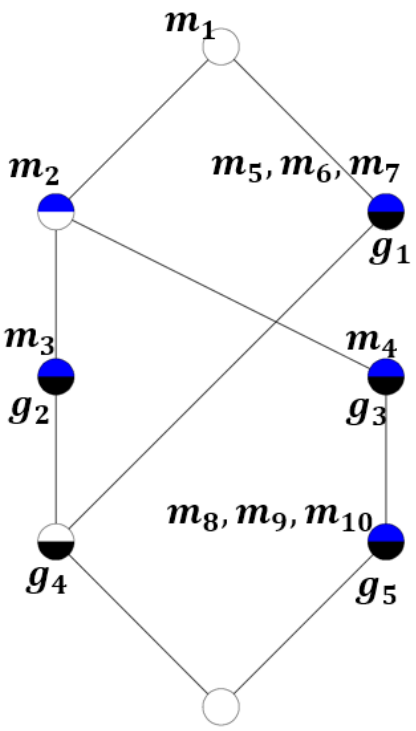


Fig. A.1 Left: The sample context to be reduced in Example 1. Right: The corresponding concept lattice of the context on the left.

introduced in Section 4.2, when applying the OR rule, an incidence x_{ij} is set to be modifiable if and only if $x_{ij} =$ FALSE and there exists another object g_k such that $x_{kj} =$ TRUE and the distance between two objects are no larger than ε_r, *i.e.*, $|(\mathrm{A}_T(g_k) - \mathrm{A}_T(g_i)) \cup (\mathrm{A}_T(g_i) - \mathrm{A}_T(g_k))| \leq \varepsilon_r$. In this example, we have:

- The distance between objects g_1 and g_4 is 2. Since $x_{12} =$ FALSE and $x_{42} =$ TRUE, we should have $c_{12} =$ TRUE. Similarly, we should have $c_{13} =$ TRUE because $x_{13} =$ FALSE and $x_{43} =$ TRUE.
- Similarly, for objects g_2 and g_3, we should have $c_{24} =$ TRUE and $c_{33} =$ TRUE.
- For any other object pairs , their distances are all beyond 2, so we have $c_{ij} =$ FALSE for any other i, j values.

From above, we can see that in this example, we have in total 4 modifiable incidences. Hence, we set 4 variables – v_{12}, v_{13}, v_{24} and v_{33}. Each variable v_{ij} represents whether the incidence x_{ij} should be modified after reduction.

After setting the variables, we are to derive the values of I_{xy}. As introduced in Section 4.3, each I_{xy} is a Boolean variable indicating whether objects g_x and g_y will be merged after reduction. In this example, we have:

- Object g_2 will be merged with g_3 if incidences x_{24} and x_{33} are both modified. That is, we should have $I_{23} = v_{24} \wedge v_{33}$.
- Object g_1 will be merged with g_4 if incidences x_{12} and x_{13} are both modified. That is, we should have $I_{14} = v_{12} \wedge v_{13}$.
- For any other object pairs, modifying the 4 modifiable incidences will not make them merge with each other, so we should have $I_{xy} = \text{FALSE}$ for any other x, y values.

Constraints for preventing too much modification

As introduced in Section 4.4, we may add the following constraint to set the maximum allowed number of modified incidences to ε_m. In this example, we have 4 modifiable incidences and the value of ε_m is 2, so we have:

$$\mathbf{1}_{v_{12}} + \mathbf{1}_{v_{13}} + \mathbf{1}_{v_{24}} + \mathbf{1}_{v_{33}} \leq 2. \tag{A.1}$$

Constraints for preventing too many inserted concepts

First, we need to compute $\text{Can}(M)$, which represent the candidate space of attribute sets that may be extent of an inserted concept. In this example, we have 4 modifiable incidences, which will introduce one attribute set, $\{m_1, m_2, m_3, m_4\}$, in $\text{Can}(M)$. Hence, suppose that this attribute set is denoted as M_1. In this example, we only need to derive the value of U_{M_1} to check whether M_1 will become extent of an inserted concept.

The value of U_{M_1} can be derived with the series of equations introduced in Section 4.5.2. First, we are to derive S_{M_1,g_i}, which represents whether $g_i \in \text{O}_{T'}(M_1)$, *i.e.*, whether g_i will have all attributes in M_1 after modification. The derivation process for S_{M_1,g_i} are as follows:

- g_4 and g_5 do not have all attributes in M_1, and both of them are unmodifiable. For g_1, it will never have the attribute m_4 since x_{14} is FALSE and unmodifiable. Hence, we should have:
$$S_{M_1,g_1} = S_{M_1,g_4} = S_{M_1,g_5} = \text{FALSE}. \tag{A.2}$$
- For g_2, if and only if the incidence x_{24} is modified will it have all attributes in M_1 and become a member of $\text{O}_{T'}(M_1)$, so we should have:
$$S_{M_1,g_2} = v_{24}. \tag{A.3}$$

- Similarly for g_3, we should have:

$$S_{M_1,g_3} = v_{33}. \tag{A.4}$$

The values of S_{M_1,g_i} can be used to derive E_{M_1}, which represents whether M_1 will become the intent of a concept after reduction. As shown by Equation (3.14), E_{M_1} will be TRUE if for any attribute $m_j \notin M_1$, there must be at least one object $g_i \in \mathrm{O}_{T'}(M_1)$ such that $x_{ij} = \text{FALSE}$ and that x_{ij} will not be modified to TRUE after reduction, *i.e.*, $\neg(c_{ij} \wedge v_{ij})$. In this example, we have two objects, g_2 and g_3, that may be elements of $\mathrm{O}_{T'}(M_1)$. It is clear that if either of them becomes an element of $\mathrm{O}_{T'}(M_1)$, E_{M_1} will be TRUE because we have $x_{ij} = \text{FALSE}$ and $c_{ij} = \text{FALSE}$ for all $i \in \{2,3\}$ and $j \in \{j \mid m_j \notin M_1\}$. However, if neither g_2 nor g_3 becomes element of $\mathrm{O}_{T'}(M_1)$, E_{M_1} will be FALSE because in this case, $\mathrm{O}_{T'}(M_1)$ will be an empty set. From above, we should have:

$$\begin{aligned} E_{M_1} &= S_{M_1,g_2} \vee S_{M_1,g_3} \\ &= v_{24} \vee v_{33}. \end{aligned} \tag{A.5}$$

Now we are to derive $F_{M_1,(G_o,M_o)}$, which represents whether this newly introduced concept of which the intent is M_1, that is, $(\mathrm{O}_{T'}(M_1), M_1)$, will be the generalization of some concept (G_o, M_o) from the original concept lattice. This should be derived with Equation (3.15). In this example, among all $(G_o, M_o) \in \mathrm{L}(T)$, four of them – $(\{g_1,g_2,g_3,g_4,g_5\},\{m_1\})$, $(\{g_2,g_3,g_4,g_5\}, \{m_1,m_2\})$, $(\{g_2,g_4\},\{m_1,m_2,m_3\})$, and $(\{g_3,g_5\},\{m_1,m_2,m_4\})$ – satisfy that $M_o \subseteq M_1$. However, none of these four concepts will satisfy that $G_o \subseteq \mathrm{O}_{T'}(M_1)$ because only g_2 and g_3 can be members of $\mathrm{O}_{T'}(M_1)$, while all four concepts have objects other than g_2 and g_3 in their extent. That is, in this example, we should have:

$$F_{M_1,(G_o,M_o)} = \text{FALSE} \quad \text{for all } (G_o, M_o) \in \mathrm{L}(T). \tag{A.6}$$

With all preliminaries derived, we can now derive U_{M_1}, that is, whether M_1 will become the intent of an inserted concept with Equation (3.16):

$$\begin{aligned} U_{M_1} &= E_{M_1} \wedge \bigwedge_{(G_o,M_o)\in \mathrm{L}(T)} \neg F_{M_1,(G_o,M_o)} \\ &= E_{M_1} \\ &= v_{24} \vee v_{33}. \end{aligned} \tag{A.7}$$

Finally, we are able to add the constraint for preventing too many inserted concepts as below:

$$\mathbf{1}_{U_{M_1}} \leq 0. \tag{A.8}$$

Target function

Now, with all variables set and all constraints added, we are to set the target function of the logical optimization problem with Equation (3.4) and run an ILP solver to get the optimal solution. In this example, the target function should be:

$$\begin{aligned} &\textbf{maximize} \quad \sum_{i=1}^{4} \mathbf{1}_{\bigvee_{i+1}^{5} I_{ij}}, \\ &\textbf{subject to} \quad (\text{A.1}) \text{ and } (\text{A.8}). \end{aligned} \tag{A.9}$$

The optimized solution to this logical optimization problem is to set v_{12} and v_{13} to be TRUE and set v_{24} and v_{33} to be FALSE. That is, to modify x_{12} and x_{13} to be TRUE and keep x_{24} and x_{33} unmodified. The reduction plan will reduce one object – g_1 and cause no inserted concept. Note that the reason why we cannot merge g_2 with g_3 by modifying x_{24} and x_{33} is that it will cause U_{M_1} to be TRUE. That is, it will introduce an inserted concept $(\{g_2, g_3\}, \{m_1, m_2, m_3, m_4\})$, which is prevented by Constraint (A.8).

A.2 Example 2: ILP-based concept lattice reduction applying the AND rule

In this example, we will use our method to reduce the formal context shown in the left panel of Fig. A.2. The corresponding concept lattice of the context is shown in the right panel of the figure:

Parameters and Variables

First, we are to determine the hyperparameters $\varepsilon_r, \varepsilon_m$ and ε_I. Here ε_r and ε_m have the exactly same meanings as in the previous example, while ε_I now represents the maximum allowed number for eliminated concepts. In this example, we set ε_r to 1, ε_m to 2 and ε_I to 3.

After setting the parameters, we are to derive the $\{c_{ij}\}$ matrix and set the variables. This step is also very similar to the case we apply the OR rule, except that now an incidence x_{ij} is set to be modifiable if and only if $x_{ij} =$ TRUE and there exists an object g_k such that $x_{kj} =$ FALSE and $|(\mathrm{A}_T(g_i) - \mathrm{A}_T(g_k)) \cup (\mathrm{A}_T(g_k) - \mathrm{A}_T(g_i))| \leq \varepsilon_r = 1$. In this example, we should have $c_{23} = c_{33} =$ TRUE and $c_{ij} =$ FALSE for any other i, j values. Hence, we set the following two variables – v_{23} and v_{33}.

With the variables set, we are now to derive I_{xy}, representing whether g_x and g_y will be merged after reduction. The derivation is completely the same as the previous example. In

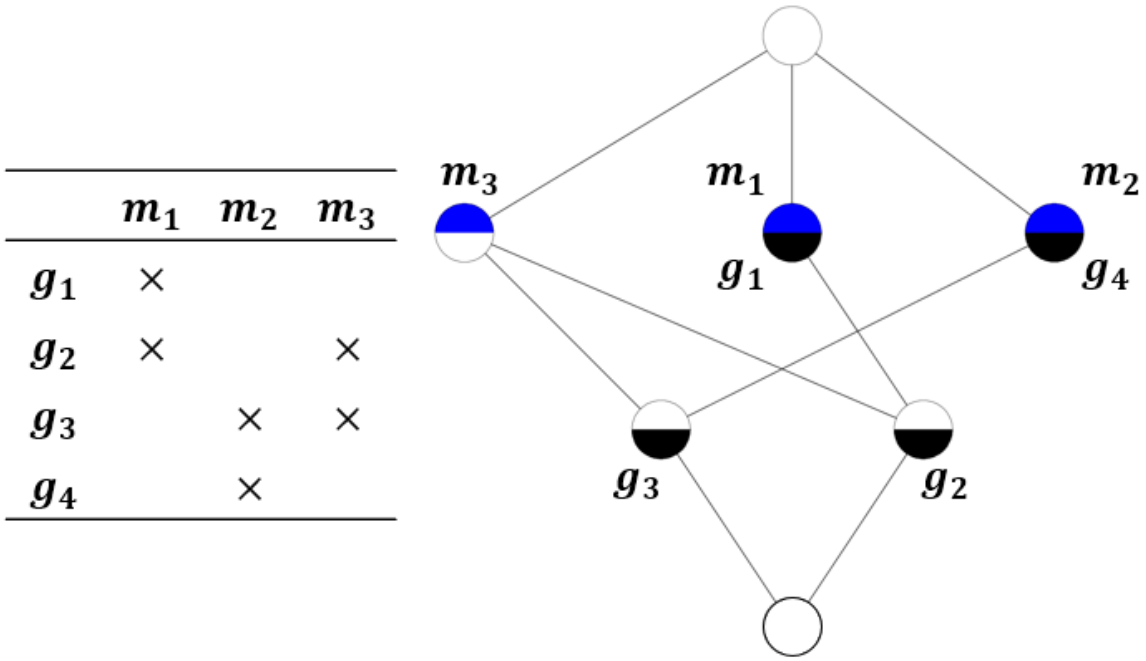


Fig. A.2 Left: The sample context to be reduced in Example 2. Right: The corresponding concept lattice of the context on the left.

this example, we should have:

$$\begin{aligned} &I_{12} = v_{23}, \\ &I_{34} = v_{33}, \\ &I_{14} = I_{23} = I_{24} = \text{FALSE}. \end{aligned} \tag{A.10}$$

Constraints for preventing too much modification

Similar to the previous example, we may add the following constraint to limit the maximum allowed number of modified incidences:

$$\mathbf{1}_{v_{23}} + \mathbf{1}_{v_{33}} \leq 2. \tag{A.11}$$

Constraints for preventing too many eliminated concepts

Now we are to derive D_{G_o,M_o}, which represents whether each concept (G_o, M_o) from the original concept lattice will be eliminated after reduction. In this example, clearly that the following four concepts – $(\{g_1, g_2\}, \{m_1\})$, $(\{g_3, g_4\}, \{m_2\})$, $(\{g_1, g_2, g_3, g_4\}, \emptyset)$ and $(\emptyset, \{m_1, m_2, m_3\})$ will not be eliminated because the former two concepts' incidences are not

modifiable and the latter two concepts have no incidence. So for these four concepts, we must have $D_{(G_o,M_o)} = \text{FALSE}$. For the other three concepts that may be eliminated, since their $D_{(G_o,M_o)}$ have completely the same derivation procedure, here we only present the derivation of $D_{(\{g_2\},\{m_1,m_3\})}$.

To get $D_{(\{g_2\},\{m_1,m_3\})}$, we first need to derive the values of $S_{\{m_1,m_3\},g_i}$, which represents whether object g_i will still have all attributes in the intent of the concept, that is, m_1 and m_3, after reduction. According to the analysis in Section 4.5.1, we only need to compute the $S_{\{m_1,m_3\},g_i}$ values for each g_i in the extent of the concept, that is, $\{g_2\}$. This is because objects not in the extent will sure have the $S_{\{m_1,m_3\},g_i}$ value to be FALSE. For the only object g_2 in the extent, it will keep having these two attributes as long as the incidence x_{23} is not modified, so we should have:

$$S_{\{m_1,m_3\},g_2} = \neg v_{23}. \tag{A.12}$$

Now we are able to derive $E_{(\{g_2\},\{m_1,m_3\})}$, which represents whether none of the objects will have all attributes in the intent of the concept, *i.e.*, m_1 and m_3, after reduction. This should be derived with Equation (3.9). In this example, we have:

$$\begin{aligned} E_{(\{g_2\},\{m_1,m_3\})} &= \bigwedge_{g_i \in \{g_2\}} \neg S_{\{m_1,m_3\},g_2} \\ &= \neg S_{\{m_1,m_3\},g_2} \\ &= v_{23}. \end{aligned} \tag{A.13}$$

Besides $E_{(\{g_2\},\{m_1,m_3\})}$, we also need to derive $F_{(\{g_2\},\{m_1,m_3\})}$, which represents whether the "featured" part of the concept will be removed after reduction. That is, whether we will have $\{m_1,m_3\} \subset \mathrm{A}_{T'}(\{g_i\})$ for all $g_i \in \mathrm{O}_{T'}(\{m_1,m_3\})$. This should be derived with Equation (3.10). In this example, this condition will never stand because there is only one object g_2 that may be a member of $\mathrm{O}_{T'}(\{m_1,m_3\})$, and the only object g_2 does not have any other attributes that are not elements of $\{m_1,m_3\}$. Hence, we should have:

$$F_{(\{g_2\},\{m_1,m_3\})} = \text{FALSE}. \tag{A.14}$$

With $E_{(\{g_2\},\{m_1,m_3\})}$ and $F_{(\{g_2\},\{m_1,m_3\})}$ derived, we are finally able to derive the value of $D_{(\{g_2\},\{m_1,m_3\})}$ with Equation (3.11) as follows:

$$\begin{aligned} D_{(\{g_2\},\{m_1,m_3\})} &= E_{(\{g_2\},\{m_1,m_3\})} \vee F_{(\{g_2\},\{m_1,m_3\})} \\ &= E_{(\{g_2\},\{m_1,m_3\})} \\ &= v_{23}. \end{aligned} \tag{A.15}$$

With $D_{(\{g_2\},\{m_1,m_3\})}$ derived, we should repeat the above procedure to derive $D_{(\{g_3\},\{m_2,m_3\})}$ and $D_{(\{g_2,g_3\},\{m_3\})}$ since we have these three concepts in the original lattice that may be eliminated. Here we directly give the results for $D_{(\{g_3\},\{m_2,m_3\})}$ and $D_{(\{g_2,g_3\},\{m_3\})}$ because the derivation processes are similar to that of $D_{(\{g_2\},\{m_1,m_3\})}$:

$$\begin{aligned} D_{(\{g_3\},\{m_2,m_3\})} &= v_{33}, \\ D_{(\{g_2,g_3\},\{m_3\})} &= v_{23} \wedge v_{33}. \end{aligned} \tag{A.16}$$

After that, we will finally be able to add the following constraint that sets the maximum allowed number of eliminated concepts to 3:

$$\mathbf{1}_{D_{(\{g_2\},\{m_1,m_3\})}} + \mathbf{1}_{D_{(\{g_3\},\{m_2,m_3\})}} + \mathbf{1}_{D_{(\{g_2,g_3\},\{m_3\})}} \leq 3. \tag{A.17}$$

With all constraints added, we can set the target function as follows and run our ILP solver to get the optimized values for the variables:

$$\begin{aligned} &\textbf{maximize} \quad \sum_{i=1}^{3} \mathbf{1}_{\bigvee_{i+1}^{4} I_{ij}}, \\ &\textbf{subject to} \quad (\text{A.11}) \text{ and } (\text{A.17}). \end{aligned} \tag{A.18}$$

The optimized solution to this logical optimization problem is to set both variables v_{23} and v_{33} to be TRUE. That is, to modify both x_{23} and x_{33} to be FALSE, which merges object g_2 with g_1, and merges object g_3 with g_4. The reduction will cause three concepts – $(\{g_3\},\{m_2,m_3\})$, $(\{g_2\},\{m_1,m_3\})$ and $(\{g_2,g_3\},\{m_3\})$ – to be eliminated, which does not exceeds the ε_I we set.

Copyright Statements

The content of the thesis contains materials from the following publications in JIP (Journal of Information Processing).

[1] Peng, S., & Yamamoto, A. (2023). Z-tca: fast algorithm for triadic concept analysis using zero-suppressed decision diagrams. *Journal of Information Processing*, 31:722-733.
DOI: https://doi.org/10.2197/ipsjjip.31.722
[2] Peng, S., & Yamamoto, A. (2024). Concept lattice reduction using integer programming. *Journal of Information Processing*, 32 [The paper will be published in September, 2024]

The content of the thesis also contains materials from the following publications in PLOS ONE. These materials apply the Creative Commons Attribution 4.0 International (CC BY) license.

[1] Peng, S., Yamamoto, A., & Ito, K. (2023). Link prediction on bipartite networks using matrix factorization with negative sample selection. *PLOS ONE*, 18(8): e0289568.
DOI: https://doi.org/10.1371/journal.pone.0289568
[2] Peng, S., Yang, H., & Yamamoto, A. (2024). BERT4FCA: A method for bipartite link prediction using formal concept analysis and BERT. *PLOS ONE*, 19(6): e0304858.
DOI: https://doi.org/10.1371/journal.pone.0304858