# Information Mining for Big Information

**Yuichi Goto**

**Abstract** Knowledge discovery of database or data (KDD) is to acquire knowledge from data. Mining interesting patterns from data or information granules is the most important process of KDD. The post-process of the mining that is to acquire knowledge from the interesting patterns is also important. As obtained interesting patterns increases, it becomes hard for analysts to do the post-process of the mining because they have done the process empirically and manually. This chapter has investigated the post-process of the mining, and presented a support method and tools for the process. Information mining is a process to acquire knowledge from the interesting patterns discovered by mining from data or information granules. Consistent verification, information abstraction, hypothesis generation, hypothesis verification, and information deduction are activities of information mining. Current data mining methods and information granulation methods are suitable for information abstract, but not suitable for the other activities. The present author has shown that strong relevant logic-based reasoning is a systematic method for supporting information mining, and introduced a forward reasoning engine, a truth maintenance system, and epistemic programming can be used for support tools of the information mining with strong relevant logic-based reasoning.

**Keywords** Information mining · Strong relevant logics · Forward reasoning engine · Truth maintenance system · Epistemic programming

Y. Goto (✉)
Department of Information and Computer Sciences, Saitama University,
Saitama 338-8570, Japan
e-mail: gotoh@mail.saitama-u.ac.jp

# 1 Introduction

The data-information-knowledge-wisdom hierarchy (DIKW hierarchy) is often used implicitly in definitions of data, information and knowledge [26]. *Data* is an elementary and recorded description of things, events, activities and transactions, lacks meaning or value, and is unorganized and unprocessed; *Information* is data processed to be meaningful, and valuable and appropriate for a specific purpose; *Knowledge* might be viewed as a mix of information and already obtained background knowledge (understanding, capability, experience, skills, values, and so on) [26]. Information is an intermediate between data and knowledge. Under the DIKW hierarchy, a process to mix obtained information with background knowledge is needed to acquire knowledge.

Knowledge discovery of database or data [15] (KDD) is to acquire knowledge from data. The word *data mining* is used for both KDD itself and a sub-process of a KDD process [19]. As a sub-process of a KDD process, data mining is a process to discover interesting patterns from massive amounts of data [15, 19]. A pattern is an expression in some language describing a subset of the data or a model applicable to the subset. An interesting pattern is a pattern that is interesting for some people in a specific domain. Interesting patterns are at least pieces of information because interesting patterns are processed data and they have meaning, from view point of the DIKW hierarchy. Interesting patterns may be adopted as pieces of knowledge directly, or they may be used as materials to acquire knowledge.

KDD with granular computing is to acquire knowledge from data via information granules. *Granular computing* is about representing, constructing, and processing information granules [27]. Informally, *information granules* can be treated as linked collections (clumps) of objects drawn together by the criteria of indistinguishability, similarity, proximity or functionality [29, 34, 35]. KDD with granular computing involves a process to discover interesting patterns from information granules [22]. Hereafter, we call the mining process *information granules mining* as like as data mining.

In the future, analysts will be able to discover the large number of interesting patterns from Big Data easily. *Big Data* is a term used to identify data sets that we cannot manage with current methodologies or software tools due to their large size and complexity [13]. Many researchers try to develop methodologies and software tools for data or information granules mining for Big Data. On the other hand, several companies and universities start to grow up data scientists [7], who are experts of KDD for Big Data. The data scientists with developed effective methodologies and tools will try to discover interesting patterns from Big Data. Moreover, the scale of Big Data and the number of kinds of data will increase [13, 24].

It is hard for analysts to acquire knowledge from the large number of interesting patterns without systematic and computer-assisted methods. In current KDD, analysts have done the process to acquire knowledge from interesting patterns empirically and manually. As the number of the interesting patterns increases, it

becomes hard to do the process empirically and manually. Supporting the process is not focused on in Big Data mining [13, 19, 24].

In this chapter, the present author has investigated the post-process of the mining, and proposed a support method and tools for the process. *Information mining* is a process to acquire knowledge from the interesting patterns discovered by mining from data or information granules, and is a post-process of the mining processes. Current data mining methods and information granulation methods are not enough for information mining. Thus, the present author has shown that strong relevant logic-based reasoning is a systematic method for supporting information mining, and introduced a forward reasoning engine, a truth maintenance system, and epistemic programming can be used for support tools of the information mining with strong relevant logic-based reasoning.

The rest of the chapter is organized as follows: Sect. 2 explains information mining and the relationship between data mining and information mining; Sect. 3 introduces strong relevant logic-based reasoning as an information mining method; Sect. 4 shows a forward reasoning engine, a truth maintenance system, and epistemic programming language as support tools for information mining with strong relevant logic-based reasoning; Sect. 5 gives a summary and future works.

## 2 Information Mining in Knowledge Discovery from Data

A KDD process typically involves data cleaning, data integration, data selection, data transformation, data mining, pattern evaluation, and knowledge presentation [19]. *Data cleaning* is a process to remove noise and inconsistent data. *Data integration* is a process to combine multiple data sources. *Data selection* is a process to retrieve data relevant to the analysis task from integrated data set. *Data transformation* is a process to transform or consolidate data into forms appropriate for mining. *Data mining* is a process to extract interesting patterns from data set. *Pattern evaluation* is a process to identify the truly interesting patterns representing knowledge. *Knowledge presentation* is a process to transform the mined knowledge into understandable representation. In [15], there is another process that is "checking for and resolving potential conflicts with previously believed (or extracted) knowledge." Here, we call the process as *consistent verification*. A KDD process is an iterative sequence of the above processes.

In KDD with granular computing, there are two other processes: information granulation and information granules mining. *Information granulation* is a process to transform or map plain data into information granules according to models based on fuzzy sets, rough sets, shadowed sets. *Information granules mining* is a process to discover interesting patterns from information granules. Figure 1 shows a control flow of processes in KDD with data mining and granular computing.

Analysts do other activities dealing with obtained interesting patterns and background knowledge after getting the interesting patterns in a KDD process. Those activities are information abstraction, hypothesis generation, hypothesis
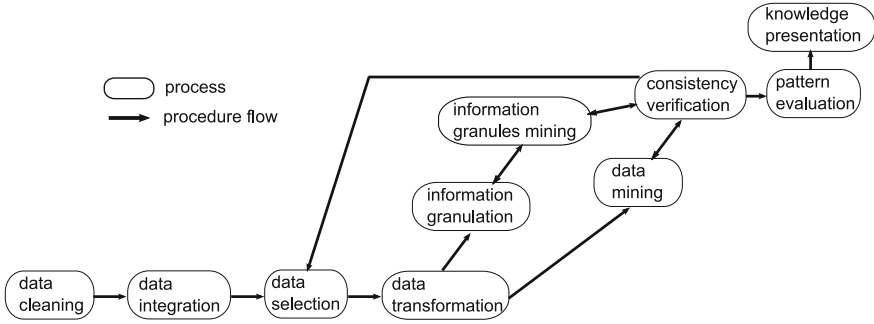
**Fig. 1** Processes in knowledge discovery from data

verification, and information deduction. *Information abstraction* is to make interesting patterns more abstract. Information depends on a context and/or a person who interprets it. While a person sees a thing as valuable information, other person may see it as data with no particular significance. Knowledge is special information for a specific purpose, but meaning and value of knowledge are widely accepted by not only a certain person, but also many people. Therefore, the degree of universality of knowledge is higher than that of information. Moreover, as the degree of universality of information (knowledge) becomes higher, the value of information (knowledge) becomes higher. Abstraction is a way to increase the degree of universality. Thus, analysts make interesting patterns more abstract. *Hypothesis generation* is to create hypotheses from the interesting patterns and background knowledge. A KDD process can be regarded as a hypothetico-deductive process. Analysts, therefore, make hypotheses not only before starting KDD, but also during KDD. *Hypothesis verification* is to verify whether already proposed hypotheses are consistent with obtained interesting patterns. Of course, analysts should verify whether already proposed hypotheses are consistent with obtained data set. In addition, they should verify the consistency of proposed hypotheses in information level. *Information deduction* is to draw previously unknown or implicitly known information from obtained interesting patterns and background knowledge. Information deduction is a way to mix obtained inter-esting patterns with background knowledge.

Those activities are related to each other. Information deduction is done before doing consistent verification and hypothesis verification. If interesting patterns or hypotheses directly conflict with already obtained interesting patterns and back-ground knowledge, analysts can find the conflicts by only checking the interesting patterns/hypotheses, already obtained interesting patterns, and background knowledge. If interesting patterns or hypotheses indirectly conflict with already obtained interesting patterns and background knowledge, analysts should deduce implicit things from the interesting patterns/hypotheses, already obtained inter-esting patterns, and background knowledge as premises, and check whether deduced things conflict with the premises. Moreover, to get more implicit infor-mation or hypotheses, information deduction is done after doing information
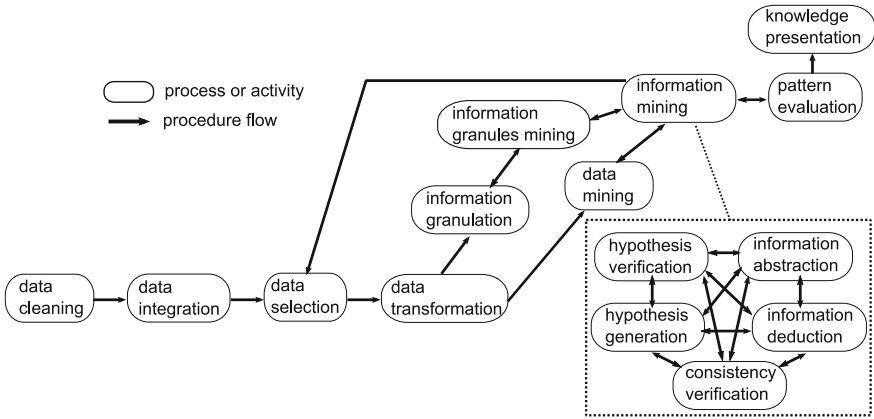
**Fig. 2** Processes in knowledge discovery from data with information mining

abstraction or hypothesis generation. Those activities are done in an arbitrary order and iteratively.

*Information mining* is a process to acquire knowledge from obtained information patterns. The activities to deal with interesting patterns and background knowledge, i.e., consistent verification, information abstraction, hypothesis generation, hypothesis verification, and information deduction are activities of information mining because the activities are used in a process to acquire knowledge from obtained information patterns. The definition of information mining is similar to García-Martínez, et. al.'s definition, i.e., "Information Mining is the sub-discipline of information systems which supports business intelligence tools to transform information into knowledge" [16]. However, they used both data mining and information mining as a same meaning, and the processes of information mining they analyzed are processes from data integration to data mining in Fig. 1. In other words, the processes are until getting interesting patterns. Both definitions are similar, but purpose and scope are different. On the other hand, the other definition of information mining [32] is data mining dealing with unstructured data, i.e., data that are not stored in databases. In data mining for Big Data, dealing with unstructured data is as a matter of course [13, 24]. The definition and the present author's definition are different. Figure 2 shows a control flow of processes in KDD with information mining.

It will be hard for analysts to do information mining without systematic and computer-assisted methods as the number of interesting patterns increases. Analysts have done the information mining empirically and manually when they try to discover knowledge from data. However, as the number of the interesting patterns increases, it becomes difficult to do information mining empirically and manually. In the future, analysts will be able to discover the large number of interesting patterns from Big Data easily as we mentioned in Sect. 1. Supporting information mining is not focused on in Big Data mining [13, 19, 24].

Current data mining methods, information granules mining methods, and information granulation methods are useful for information abstraction, but not enough to do the other activities in information mining. The purpose of current data mining methods is to help pattern discovery [19, 33]. To discover patterns is a way of information abstraction. Information granulation is also a way of information abstraction. Thus, we can use data mining methods, information granules mining methods, and information granulation methods for information abstraction. However, they cannot be applied to the other activities of the information mining: consistency verification, hypothesis generation, hypothesis verification, and information deduction.

Requirements of hopeful information mining methods are as follows. (1) The method should be able to deal with all of information mining activities because analysts should do not only one activity but also several activities several times. (2) The method can be done automatically or semi-automatically. (3) Formalization method to describe interesting patterns and background knowledge in the method should have enough expressiveness because the outputs of the data mining methods and information granules mining methods are represented as various forms.

## 3 Strong Relevant Logic-Based Reasoning as an Information Mining Method

### 3.1 Deduction, Induction, and Abduction in Information Mining

Reasoning is helpful for consistency verification, information abstraction, hypothesis generation, hypothesis verification, and information deduction. Reasoning is the process of drawing new conclusions from given premises, which are already known facts or previously assumed hypotheses (Note that how to define the notion of new formally and satisfactorily is still a difficult open problem until now) [3, 4].

Reasoning can be classified into three forms, deductive reasoning, inductive reasoning, and abductive reasoning. Deductive reasoning (deduction) is the process of deducing or drawing conclusions from some general principles already known or assumed. Inductive reasoning (induction) is the process of inferring some general laws or principles from the observation of particular instances. Abductive reasoning (abduction) is the process whereby a surprising fact is made explicable by the application to it of a suitable proposition. The deductive reasoning guarantees that the conclusions deduced in the process are true if all premises are true, but inductive and abductive reasoning do not do that.

Deductive reasoning plays an important role for consistency verification and hypothesis verification. To do both verifications, it is necessary to check whether contradictions are drawn from discovered interesting patterns/given hypotheses

and background knowledge. Analysts cannot know what contradictions occur so that it is difficult and ad hoc to check that proposition by using proving. By using deductive reasoning, analysts can check the proposition systematically. Doing information deduction is just deductive reasoning. Inductive reasoning is a way of information abstraction. Similarly, abductive reasoning is a way of hypothesis generation.

## 3.2 Formal Logic System and Formal Theory

To accept results of consistency verification, hypothesis generation, and information deduction with deductive reasoning as correct ones, deductive reasoning should be a logically valid reasoning. When doing deductive reasoning, deduced conclusions should be true if premises are true, and that the conclusions should be related to the premises. How can we ensure such feature of deductive reasoning? The answer is logics. A logically valid reasoning is a reasoning such that its arguments are justified based on some logical validity criterion provided by a logic system in order to obtain correct conclusions (Note that here the term correct does not necessarily mean true) [3, 4].

In general, a formal logic system $L$ consists of a formal language, called the object language and denoted by $F(L)$, which is the set of all well-formed formulas of $L$, and a logical consequence relation, denoted by meta-linguistic symbol $\vdash_L$, such that $P \subseteq F(L)$ and $c \in F(L)$, $P \vdash_L c$ means that within the frame work of $L$, $c$ is valid conclusion of premises $P$, i.e., $c$ validly follows from $P$. For a formal logic system $(F(L), \vdash_L)$, a logical theorem $t$ is a formula of $L$ such that $\phi \vdash_L t$ where $\phi$ is empty set. Let $Th(L)$ denote the set of all logical theorems of $L$. $Th(L)$ is completely determined by the logical consequence relation $\vdash_L$. According to the representation of the logical consequence relation of a logic, the logic can be represented as a Hilbert style axiomatic system, a Gentzen natural deduction system, a Gentzen sequent calculus system, or other type of formal system. A formal logic system $L$ is said to be *explosive* if and only if $\{A, \neg A\} \vdash_L B$ for any two different formulas $A$ and $B$; $L$ is said to be *paraconsistent* if and only if it is not explosive.

A formal theory with premises $P$ based on $L$, called a $L$-theory with premises $P$ and denoted by $T_L(P)$, is defined as $T_L(P) =_{df} Th(L) \cup Th_L^e(P)$, and $Th_L^e(P) =_{df} \{et \mid P \vdash_L et \text{ and } et \neq Th(L)\}$ where $Th(L)$ and $Th_L^e(P)$ are called the logical part and the empirical part of the formal theory, respectively, and any element of $Th_L^e(P)$ is called an empirical theorem of the formal theory. Figure 3 shows the relationship among $F(L)$, $T_L(P)$, $Th(L)$, and $Th_L^e(P)$ of a formal logic system $L$. A formal theory $T_L(P)$ is said to be *directly inconsistent* if and only if there exists a formula $A$ of $L$ such that both $A \in P$ and $\neg A \in P$ hold. A formal theory $T_L(P)$ is said to be *indirectly inconsistent* if and only if it is not directly inconsistent but there exists a formula $A$ of $L$ such that both $A \in T_L(P)$ and $\neg A \in T_L(P)$; a formal theory $T_L(P)$ is said to be *consistent* if and only if it is neither directly inconsistent
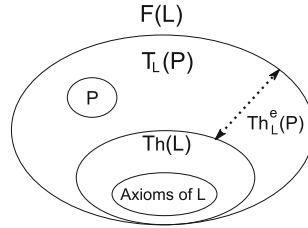
**Fig. 3** *L*-theory with premises *P*

nor indirectly inconsistent. A formal theory $T_L(P)$ is said to be explosive if and only if $A \in T_L(P)$ for any $A \in F(L)$; $T_L(P)$ is said to be *paraconsistent* if and only if it is not explosive. An explosive formal theory is not useful at all. Therefore, any meaningful formal theory should be paraconsistent [3, 4]. Note that if a formal logic system *L* is explosive, then any directly or indirectly inconsistent *L*-theory $T_L(P)$ must be explosive.

## 3.3 Reasoning with Strong Relevant Logics

To do deductive reasoning based on a formal logic *L* for information mining is to obtain $Th_L^e(P)$ where *P* is premises that represent already discovered patterns given by the mining activities. Today, there are so many different logic systems motivated by various philosophical considerations. As a result, a reasoning may be valid for one logical validity criterion but invalid for another. If logic systems underlying deductive reasoning are different, results of deductive reasoning may be different. In other words, although premises *P* is same, $Th_L^e(P)$ and $Th_{L'}^e(P)$ may be different where formal logic system *L* and $L'$ are different. Thus, we have to choose a suitable logic system underlying deductive reasoning.

A logic system underlying deductive reasoning should ensure truth-preserving, relevant, ampliative, paracomplete, paraconsistent reasoning [3, 4]. Strong relevant logic and its family [3, 4] are hopeful candidates for logic systems underlying deductive reasoning in information mining process. Classical mathematical logic (CML for short) has been widely used for logic system underlying proof and reasoning. However, reasoning based on CML and its conservative extensions is truth-preserving, but not relevant, ampliative, and paraconsistent [1–4]. Thus, CML and its conservative extensions are not suitable for logic system underlying reasoning in information mining process. Relevant logics were constructed as logic systems that are more suitable for underlying reasoning rather than CML and its extensions [1, 2]. After that, as logic systems that are more suitable for underlying reasoning rather traditional relevant logics, strong relevant logics [3] were proposed. Then, family of strong relevant logics, e.g., temporal relevant logics, deontic relevant logics, spatial relevant logics, were also proposed [4]. Reasoning

based on strong relevant logics and its family is truth-preserving, relevant, ampliative, paracomplete, and paraconsistent reasoning.

Meanwhile, logic-based formalization is suitable for representing already discovered interesting patterns in data mining or information granules mining. Basically, the discovered patterns denote the relationship among data objects, discovered classes, and the target data set [17]. Logic-based representation is suitable for describing such qualitative information. Especially, conditional relations, i.e., "if … then …," are useful for describing above relations. By using logic-based formalization, we can ignore the difference of approaches among data mining methods and information granules mining methods that are used to obtain patterns when we try to do information mining. Moreover, we can deal with background knowledge as well as discovered interesting patterns by using logic-based formalization.

Consequently, we can conclude that strong relevant logic-based reasoning is hopeful as a systematic method of the information mining. Strong relevant logic-based reasoning means (1) adopting strong relevant logics or its family as a logic system underlying reasoning, (2) formalizing target information into logical formulas based on the logic system, and (3) doing deductive, inductive, and abductive reasoning based on the logic system.

# 4 Supporting Tools for Information Mining with Strong Relevant Logic-Based Reasoning

## 4.1 Forward Reasoning Engine

As mentioned above in Sect. 3, to do deductive reasoning based on a formal logic $L$ for information mining is to obtain a set of all empirical theorems $Th_L^e(P)$ where $P$ is premises that represent already discovered patterns given by the mining activities; and empirical theorems are theorems in a target domain. Is there a support tool to obtain the $Th_L^e(P)$? That is a forward reasoning engine.

A forward reasoning engine is a computer program to automatically draw new conclusions by repeatedly applying inference rules, which are programmed in the reasoning engine or given by users to the reasoning engine as input, to given premises and obtained conclusions until some previously specified conditions are satisfied. The first forward reasoning engine is "Logic Theory Machine," developed by Newell, Shaw and Simon in 1957. As a well-known fact, the Logic Theory Machine was not successful due to the problem of computational complexity [8]. This (and the resolution method discovered by Robinson) led almost all researchers to adopt the more efficient approach of backward reasoning but not approach of forward reasoning [25]. However, from the viewpoint of logic validity of reasoning, the failure of Logic Theory Machine is caused by classical mathematical logic rather than forward reasoning [6] as mentioned above in Sect. 3. If

we want to create, discover, or predict some new things rather that prove some things previously specified, then the only way is to ask forward reasoning. Forward reasoning engines support analysts to information mining activities by doing strong relevant logic-based reasoning automatically or semi-automatically.

FreeEnCal [6, 18] was proposed and developed as a forward reasoning engine with general-purpose, and is a hopeful candidate for a forward reasoning engine for information mining with strong relevant logic-based reasoning. It can interpret specifications written in the formal language such that any user can use the formal language to describe and represent formulas and inference rules for deductive, simple inductive, and simple abductive reasoning. It also can reason out all or a part of logical theorem schemata of a logic system, i.e., $Th(L)$ where $L$ is a formal logic system as mentioned in Sect. 3, under the control conditions attached to the reasoning task specified by users, and all or a part of empirical theorems of a formal theory and facts, $Th_L^e(P)$ where $P$ is premises, under the control conditions attached to the reasoning task specified by users. We can adopt FreeEnCal as a support tool for doing information mining with strong relevant logic-based reasoning.

## 4.2 Truth Maintenance System

An information mining process must be non-monotonic. In general, obtained interesting patterns and background knowledge may be incomplete and inconsistent. Moreover, inductive and abductive reasoning do not guarantee that the drawn conclusions are true if all premises are true. Thus, as information mining progresses, the amount of information may change because of solving contradictional information or reducing old or wrong information in the target cluster of information.

A truth maintenance system [10] (TMS for short), and also called a belief revision system or reason maintenance system was proposed to realize information systems to deal with such non-monotonic processes. A TMS works with an inference engine. The inference engine is a program to draw derived data from premises, assumptions, and other derived data, and it gives the derived data to the TMS (e.g., a forward reasoning engine is a kind of inference engines). *Premises* are used to define data that is always true. This data is not dependent on other data, and is not inferred from other facts. *Assumptions* are believed in the lack of evidence to the contrary, and are taken to be true until the contrary is proved. Premises, assumptions, and derived data managed in a TMS are called *beliefs*. If the TMS detects a contradiction in the current belief set stored in it, then the TMS eliminates it by revising the current belief set. When the TMS is revising the current belief set, it uses justifications of derived data for searching which assumptions are causes of the contradiction. *Justifications* describe the dependencies between data. The TMS gives all beliefs in the current belief set to the

inference engine when the engine requires them. The main task of TMSs is to keep consistency of the current belief set stored in the TMSs.

The first TMS was proposed by Doyle [10]. After that, many TMSs were proposed. Stanojevic et al. [30] classified TMSs into three kinds: justification-based TMS (JTMS) [10], assumption-based TMS (ATMS) [9], and logic-based TMS (LTMS) [20, 21]. JTMSs and LTMSs can deal with only one context while ATMSs can deal with multi-context. A *context* is a set of all data that can be derived from an environment. An *environment* is a set of assumptions that uniquely describe a state. One context is uniquely determined by the corresponding environment. An environment is consistent if a contradiction cannot be inferred from the corresponding set of assumptions. To find which environment is inconsistent, ATMSs use labels. A *label* can be attached to each datum, describing which environment it will hold in. Label contains sets of environments in which the corresponding facts are valid. Using labels, we can immediately tell whether or not a datum holds under some assumptions. A *node* represents a data structure that usually contains an index (used to describe the node uniquely), a corresponding inference engine's datum, its justification (or justifications), and a label (in ATMSs and LTMSs). JTMSs and ATMSs do not require that premises, assumption, and derived data are represented as logical formulas while LTMSs require that to provide a facility of proof by refutation without inference engines. There are several extensions of ATMSs that focus on the uncertainty of assumptions [11, 12, 23, 28].

TMSs are a useful mechanism to support information mining with strong relevant logic-based reasoning. In the information mining, premises are background knowledge of a target domain and problem; assumptions are already discovered interesting patterns by mining activities, and drawn pieces of information by inductive reasoning or abductive reasoning; an inference engine is a forward reasoning engine that can deal with strong relevant logic-based reasoning. By using TMSs, it is possible to manage the consistency of the current set of pieces of information automatically or semi-automatically.

However, the above traditional TMSs are not suitable for cooperating with inference engines that do reasoning based on paraconsistent logics like strong relevant logics [17]. An operation to keep a consistency of current belief set is a primitive operation for traditional TMSs. Logic systems underlying traditional TMSs are classical mathematical logic (CML) or its conservative extensions. Reasoning based on those logics is inconsistent reasoning, i.e., the reasoning allows that everything follows from a contradiction. Thus, traditional TMSs should solve the inconsistency of the current belief set as soon as possible when contradictions are found in the belief set. Unlike reasoning based on CML and its conservative extensions, reasoning based on strong relevant logics and its family is paraconsistent. It allows that contradictions are in premises. An operation to keep a consistency of current belief set is not a primitive operation of TMSs for paraconsistent reasoning. There is a gap between traditional TMSs and TMSs for information mining with relevant logic-based reasoning. The present author has proposed the TMS for paraconsistent reasoning [17], and has been developing it.

## 4.3 Epistemic Programming

As the supporting tools, it will be necessary to prepare an environment to simulate epistemic processes that include a process of elimination, process of reduction to absurdity, and processes of deductive reasoning, inductive reasoning, and abductive reasoning by using a forward reasoning engine and a truth maintenance system. Information mining is not easy task as well as data mining and information granules mining because analysts do not know what information or knowledge there is in obtained interesting patterns and how they can extract such useful information or knowledge from the interesting patterns and background knowledge. The analysts may acquire the information or knowledge by trial and error. Therefore, supporting tools for reducing the cost of information mining will be demanded.

Epistemic programming [3] and its programming language [14, 31] can be used for constructing such simulation environment. A strong relevant logic model of epistemic processes in scientific discovery, and Epistemic programming was proposed as a novel program paradigm to program epistemic processes in scientific discovery.

Let $T_L(K)$ be an $L$-theory with premises $K$ where $K \subseteq F(L)$ is a set of sentences to represent the explicitly known knowledge and/or current beliefs of an agent. An explicitly epistemic operation by the agent is any one of the following operations: for any $A \in T_L(K) - K$ where $T_L(K) \neq K$, an explicitly epistemic deduction of $A$ from $K$, denoted by $K^{d+A}$, is defined as $K^{d+A} =_{df} K \cup \{A\}$; for any $A \notin T_L(K)$ (note that we do not require $\neg A \notin T_L(K)$), an explicitly epistemic expansion of $K$ by $A$, denoted by $K^{e+A}$, is defined as $K^{e+A} =_{df} K \cup \{A\}$, in particular, an explicitly epistemic simple-induction is an explicitly epistemic expansion $K^{e+\forall x(A)}$ for $\exists x(A) \in K$ and an explicitly epistemic abduction is an explicitly epistemic expansion $K^{e+A}$ for $C \in K$ and $A \Rightarrow C \in K$ where $\Rightarrow$ denotes the notion of implication (entailment) in $L$; for any $A \in K$, an explicitly epistemic contraction of $K$ by $A$, denoted by $K^{-A}$, is defined as $K^{-A} =_{df} K - \{A\}$, in particular, an explicitly epistemic consistent-contraction is an explicitly epistemic contraction $K^{-A}$ for $\neg A \in K$ or $K - \{\neg A\}$ for $A \in K$.

Let $T_L(K)$ be an $L$-theory with premises $K$ where $K \subseteq F(L)$ is a set of sentences to represent the explicitly known knowledge and/or current beliefs of an agent. An implicitly epistemic operation by a forward reasoning engine (e.g., FreeEnCal) is any one of the following operations: or any $K$, an implicitly epistemic deduction of $K$, denoted by $K^d$, is defined as $K^d =_{df} T_L(K)$; for any $A \notin T_L(K)$ (note that we do not require $\neg A \notin T_L(K)$), an implicitly epistemic expansion of $K$ by $N$ to deduce $A$, denoted by $T_L(K \cup N)^{e+A}$, is defined as $T_L(K \cup N)^{e+A} =_{df} T_L(K \cup N)$ where $N \subseteq F(L)$ such that $A \notin T_L(K)$ but $A \in T_L(K \cup N)$; in particular, an implicitly epistemic simple-induction is an implicitly epistemic expansion $T_L(K \cup N)^{e+\forall x(A)}$ for $\exists x(A) \in T_L(K)$ and an implicitly epistemic abduction is an implicitly epistemic expansion $T_L(K \cup N)^{e+\forall x(A)}$ for $C \in T_L(K)$ and $A \Rightarrow C \in T_L(K)$ where $\Rightarrow$ denotes the notion of implication (entailment) in $L$; for any $A \in T_L(K)$, an implicitly

epistemic contraction of $K$ by $N$ to delete $A$, denoted by $T_L(K-N)^{-A}$, is defined as $T_L(K-N)^{-A} =_{df} T_L(K-N)$ where $N \subseteq K$ such that $A \notin T_L(K-N)$, in particular, an implicitly epistemic consistent-contraction is an implicitly epistemic contraction $T_L(K-N)^{-A}$ for $\neg A \in T_L(K)$ or $T_L(K-N)^{-A}$ for $A \in T_L(K)$.

Thus, an epistemic process of deductive-inductive-abductive belief revision can be defined as a sequence $K_0$, $o_1$, $K_1$, $o_2$, $K_2$, ..., $K_{n+1}$, $o_n$, $K_n$ where $K_i \subseteq F(L)$ $(0 \le i \le n)$, called an epistemic state of the epistemic process, is a set of sentences to represent known knowledge and current beliefs of an agent, and $o_{i+1}$ $(0 \le i \le n)$, is any of explicitly or implicitly epistemic operations, and $K_{i+1}$ is the result of applying $o_{i+1}$ to $K_i$. In particular, $K_0$ is called the primary epistemic state of the epistemic process, and $K_n$ is called the terminal epistemic state of the epistemic process, respectively.

Note that the above definitions of epistemic operations and epistemic processes are general but not dependent on any special logic system. In [3], strong relevant logics are used for logic systems underlying the strong relevant logic model. Then, temporal relevant logics [4] are used for the model [5]. We can choose a suitable logic system in a family of strong relevant logics as a logic system underlying the model according to a target problem.

An epistemic program is a sequence of instructions such that for a primary epistemic state given as the initial input, an execution of the instructions produces an epistemic process where every epistemic operation corresponds to an instruction whose execution results in an epistemic state, in particular, the terminal epistemic state is also called the result of the execution of the program. We say that an epistemic program *replays* a scientific discovery if the execution of the program produces the same result as that discovered by the original discoverer in history when the program as input takes the same initial conditions as the original discoverer did. We say that an epistemic program *creates* or *makes* a scientific discovery if the execution of the program produces a result that is new, important, and interesting to the scientists working on the particular domain under investigation. An information mining process can be regarded as an epistemic process of scientific discovery process. By using Epistemic programming and its programming language, analysts can construct an environment to simulate or help to do own information mining processes.

Study of Epistemic programming is ongoing work. EPLAS was proposed as the first epistemic programming language [31], and its implementation was also proposed and developed [14]. A forward reasoning engine and a truth maintenance system are parts of the implementation of EPLAS. However, current EPLAS and its implementation provide poor representation power. For example, the current EPLAS and its implementation do not provide scientists with a high-level and general-purpose mechanism to deal with belief revision [17]. As a result, users of EPLAS have to program their belief revision processes by primary epistemic operations. That is not an easy task. To use EPLAS and its implementation for constructing the support environment for information mining, it is necessary to enrich representation power of them.

# 5  Summary

The chapter has investigated information mining as a new challenging issue in knowledge discovery from data (KDD) for Big Data. Information mining is a process to acquire knowledge from interesting patterns discovered by the mining from data or information granules, and is a post-process of the mining processes in a KDD process. Consistent verification, information abstraction, hypothesis generation, hypothesis verification, and information deduction are activities of information mining. In current KDD, analysts have done the information mining empirically and manually. However, it will be hard to do the information mining without systematic and computer-assisted methods as the number of interesting patterns increases. Current data mining methods and information granulation methods are suitable for information abstract, but not suitable for the other activities of information mining. The chapter has shown that strong relevant logic-based reasoning is a systematic method for supporting information mining, and introduced a forward reasoning engine, a truth maintenance system, and epistemic programming can be used for support tools of the information mining with strong relevant logic-based reasoning.

This is an ongoing work. Current epistemic programming language and its implementation are not enough to support for information mining. To improve and implement them are future works. Information granulation is a good way of information abstraction. Thus, to integrate information granulation and strong relevant logic-based reasoning is also a future work.

# References

1. Anderson, A.R., Belnap Jr, N.D.: Entailment: the Logic of Relevance and Necessity, vol. 1. Princeton University Press, Princeton (1975)
2. Anderson, A.R., Belnap Jr, N.D., Dunn, J.M.: Entailment: the Logic of Relevance and Necessity, vol. 2. Princeton University Press, Princeton (1992)
3. Cheng, J.: A strong relevant logic model of epistemic processes in scientific discovery. In: Kawaguchi, E., Kangassalo, et al. (eds.) Information Modeling and Knowledge Bases XI. Frontiers in Artificial Intelligence and Applications, vol. 61, pp. 136–159. IOS Press, Amsterdam (2000)
4. Cheng, J.: Strong relevant logic as the universal basis of various applied logics for knowledge representation and reasoning. In: Kiyoki, Y., et al. (eds.) Information Modeling and Knowledge Bases XVII. Frontiers in Artificial Intelligence and Applications, vol. 136, pp. 310–320. IOS Press, Amsterdam (2006)
5. Cheng, J.: A temporal relevant logic approach to modeling and reasoning about epistemic processes. In: 2009 fifth international conference on semantics, knowledge and grids (SKG 2009), pp. 19–25. IEEE Computer Society, Beijing (2009)
6. Cheng, J., Nara, S., Goto, Y.: FreeEnCal: a forward reasoning engine with general-purpose. In: Apolloni, B., Howlett, R.J., Jain, L.C. (eds.) Knowledge-Based Intelligent Information and Engineering Systems, 11th International Conference, KES 2007, XVII Italian Workshop on Neural Networks, Vietri sul Mare, Italy, 12–14, Sept 2007. Proceedings, Part II. Lecture

Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science), vol. 4693, pp. 444–452. Springer, Berlin (2007)

7. Davenport, T.H., Patil, D.J.: Data scientist: the sexiest job of the 21st century. Harvard Bus. Rev. **90**(10), 70–77 (2012)

8. Davis, M.: The early history of automated deduction. In: Robinson, A., Voronkov, A. (eds.) Handbook of Automated Reasoning, pp. 5–15. Elsevier and MIT Press, Amsterdam/ Cambridge (2001)

9. de Kleer, J.: An assumption-based TMS. Artif. Intell. **28**(2), 127–162 (1986)

10. Doyle, J.: A truth maintenance system. Artif. Intell. **12**(3), 231–272 (1979)

11. Dubois, D., Berre, D.L., Prade, H., Sabbadin, R.: Using possibilistic logic for modeling qualitative decision: ATMS-based algorithms. Fundamenta Informaticae **37**(1–2), 1–30 (1999)

12. Dubois, D., Lang, J., Prade, H.: Handling uncertain knowledge in an ATMS using possibilistic logic. In: Ras, Z., Emrich, M. (eds.) Methodologies for Intelligent Systems, 5: Proceedings of the 5th International Symposium on Methodologies for Intelligent Systems Held 25–27 Oct 1990, pp. 252–259. Elsevier, Amsterdam (1990)

13. Fan, W., Bifet, A.: Mining big data: current status, and forecast to the future. ACM SIGKDD Explor. Newslett. **14**(2), 1–5 (2012)

14. Fang, W., Takahashi, I., Goto, Y., Cheng, J.: Practical implementation of EPLAS: an epistemic programming language for all scientists. In: 2011 international conference on machine learning and cybernetics (ICMLC 2011), pp. 608–616. IEEE, Guilin (2011)

15. Fayyad, U., Piatetsky-shapiro, G., Smyth, P.: From data mining to knowledge discovery in databases. AI Mag. **17**(3), 37–54 (1996)

16. García-Martínez, R., Britos, P., Rodríguez, D.: Information mining processes based on intelligent systems. In: Ali, M., Bosse, T., Hindriks, K., Hoogendoorn, M., Jonker, C., Treur, J. (eds.) Recent Trends in Applied Artificial Intelligence, 26th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2013, Amsterdam, The Netherlands, 17–21 June 2013. Proceedings. Lecture Notes in Computer Science, vol. 7906, pp. 402–410. Springer, Berlin (2013)

17. Goto, Y., Cheng, J.: A truth maintenance system for epistemic programming environment. In: 2012 eighth international conference on semantics, knowledge and grids (SKG 2012), pp. 1–8. IEEE Computer Society, Beijing (2012)

18. Goto, Y., Koh, T., Cheng, J.: A general forward reasoning algorithm for various logic systems with different formalizations. In: Lovrek, I., Howlett, R.J., Jain, L.C. (eds.) Knowledge-Based Intelligent Information and Engineering Systems, 12th International Conference, KES 2008, Zagreb, Croatia. 3–5 Sept 2008. Proceedings, Part II. Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science), vol. 5178, pp. 526–535. Springer, Berlin (2008)

19. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann Publishers, Burlington, MA (2011)

20. McAllester, D.A.: An Outlook on Truth Maintenance. AI Memos 551 (1980)

21. McDermott, D.: A general framework for reason maintenance. Artif. Intell. **50**(3), 289–329 (1991)

22. Mitra, S., Pal, S.K., Mitra, P.: Data mining in soft computing framework: a survey. IEEE Trans. Neural Netw. **13**(1), 3–14 (2002)

23. Monai, F.F., Chehire, T.: Possibilistic assumption based truth maintenance system, validation in a data fusion application. In: The eighth international conference on uncertainty in artificial intelligence (UAI92), pp. 83–91. Morgan Kaufmann Publishers, Stanford, CA (1992)

24. O'Leary, D.E.: Artificial intelligence and big data. IEEE Intell. Syst. **28**(2), 96–99 (2013)

25. Robinson, A., Voronkov, A. (eds.): Handbook of Automated Reasoning, vol. 1–2. Elsevier and MIT Press, Amsterdam/Cambridge (2001)

26. Rowley, J.: The wisdom hierarchy: representations of the DIKW hierarchy. J. Inf. Sci. **33**(2), 163–180 (2007)

27. Pedrycz, W.: Granular Computing: Analysis and Design of Intelligent Systems. CRC Press/Taylor & Francis, Boca Roton (2013)
28. Shen, Q., Zhao, R.: A credibilistic approach to assumption-based truth maintenance. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **41**(1), 85–96 (2011)
29. Skowron, A., Stepaniuk, J.: Information granules: towards foundations of granular computing. Int. J. Intell. Syst. **16**(1), 57–85 (2001)
30. Stanojevic, M., Vranes, S., Velasevicngine, D.: Using truth maintenance systems: a tutorial. IEEE Intell. Syst. **9**(6), 46–56 (1994)
31. Takahashi, I., Nara, S., Goto, Y., Cheng, J.: EPLAS: an epistemic programming language for all scientists. In: Shi, Y. (ed.) Computational Science—ICCS 2007: 7th International Conference, Beijing, China. 27–30 May 2007. Proceedings, Part I. Lecture Notes in Computer Science, vol. 4487, pp. 406–413. Springer, Berlin (2007)
32. Tkach, D.S.: Information mining with the IBM intelligent miner family. In: An IBM Software Solutions White Paper. pp. 1–29. IBM (1998)
33. Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., et al.: Top 10 algorithms in data mining. Knowl. Inf. Syst. **14**(1), 1–37 (2008)
34. Zadeh, L.: Fuzzy logic = computing with words. IEEE Trans. Fuzzy Syst. **4**(2), 103–111 (1996)
35. Zadeh, L.: Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. Fuzzy Sets Syst. **90**(2), 111–127 (1997)