A Rule-Map based Technique for Information Inconsistency Verification

Jun Ma $^{\#}$, Jie Lu, Guangquan Zhang
Faculty of Information Technology, University of Technology, Sydney
P.O. Box 123, Broadway, NSW 2007, Australia
{*junm, jielu, zhangg*}*@it.uts.edu.au*

Abstract

This paper focuses on the problem of verifying information inconsistencies in acquired information. A rule-map based technique for data inconsistency is presented, where rule-map is used to describe hierarchical structure of rules and estimate judgment standard for consistency dynamically. Moreover, a state-based knowledge representation technique for logical inconsistency is investigated, in which knowledge is illustrated as states set of related objects and logical inconsistency is determined by the relationships between those state-sets. To illustrate the presented techniques, two examples are given.

1. Introduction

Intelligent systems have been widely used in various areas, such as industrial control, business management, medical diagnose, market analysis, public security, as assistant tools for supporting decision making [5]. For the sake of designing and developing an intelligent system, a combination of models, methods, and techniques for effective information processing is employed [5]. Generally speaking, information processing consists of a series of processing stages, including information acquisition, representation, verification, filtration, integration, and application. Among them, information verification plays an important role, which can assure the reliability of information to be processed in the following steps. Because a knowledge base is required when we develop an intelligent system for a specific domain, this paper mainly focuses on information inconsistency verification based on knowledge base.

Information inconsistency verification is referred to as two aspects: (1) how to find the inconsistencies of acquired information, which consists of data inconsistency and logical inconsistency, and (2) how to find the anomalies of knowledge base, which includes redundancy, conflict, circularity, and deficiency [13]. Methods and systems for the second aspects are widely investigated, but few on the first aspect. This paper aims at verifying the inconsistency of acquired information.

There may be two kinds of inconsistencies in the acquired information. One occurs when the information is not compatible with the main part of the information acquired, which is called data inconsistency. The other one occurs when using acquired information, it may lead to wrong conclusions but each piece of information could be right. The second type of inconsistency is called logical inconsistency. In this paper, we shall present a model to verify these two kinds of inconsistencies.

The rest of the paper is organized as follows. Section 2 reviews some related works. In Section 3, a rule-map based method for verifying data inconsistency is illustrated. A state-based method for verifying logical inconsistency is described in Section 4. In Section 5, two examples are given to illustrate the presented methods. Our future work is discussed in Section 6.

2. Related works

Information verification techniques were developed rapidly in the past decade in data base and expert systems. Most existing works are concentrated on the verification of knowledge base (rule base) in order to reduce errors and faults during system development and maintenance. Roughly speaking, these approaches can be divided into two classes:

1) Approaches based on logical inference or logical deduction.
2) Approaches based on graph-based searching or model matching.

Because first-order predicated logic is a powerful knowledge representation method, logical inference based approaches play important role in information verification. Polat [12] applied unification of rules to verify anomalies in knowledge base. Mazure et al [7] used the bound resolution and the local searching method for satisfiability problem to knowledge base verification. Because SAT problem is NP-hard and semi-determined, their methods is hardly to be applied to real problem directly. Wu et al [15] discussed how to find and modify the inconsistency based on resolution principle. Zhang et al [18] presented an illustration for inconsistency, redundancy, circularity, and incompleteness in terms of theories in the first order predicate logic. Moreover, Zhang et al [18] pointed out that without a clear semantics, it is hard to formally define and analyze knowledge base anomalies and to assess the effectiveness of verification tools, methods, and techniques.

Graph-based searching technique including methods based on Petri nets and its extensions, binary directed graph, or others, is another important strategy to verify inconsistency in knowledge base. Graph-based searching methods describe the connection between conditions and conclusions of rules by edges between nodes in a directed graph or an incident matrix.

Park and Seong [9] reported a method based on extended colored Petri nets for knowledge acquisition and verification for nuclear power plant dynamic alarms. In their method, inconsistent rules are described by different sub graphs, each of them corresponding to a kind of inconsistency, and the verifying task is realized by searching those sub graphs. Yang et al [17] proposed a Petri nets formalism for verifying rule-based systems, in which the rules are Horn clauses form. Their method includes three phases: rule normalization, rule transformation, and rule verification. The rule verification phase has two Petri nets models, i.e., the static analysis and dynamic simulated Petri nets model. Moreover, they applied these models to verify inconsistency for fuzzy rules. Botten [2] described rules in a matrix, which is similar to the incident matrix in Petri nets theory. In Botten's method, the verification of knowledge base is carried out by simple operations on the constructed matrix. Similarly, Mues et al [8] discussed a method based on binary decision diagram, in which rules' input space are encoded into binary forms, and rule verification is checking whether some resulting labels are against each other. This method is more like semantical resolution in SAT problem.

To our best knowledge, aforementioned strategies have some drawbacks in the following aspects.

(1) From the point of view of logic, predicated logic is suitable to describe universal knowledge and, in general, it has only two possible values. However, in real problem, the applied knowledge is always domain specified and the underlying logic may be many-valued logic. Hence, first-order predicated logic based verification methods cannot be the best choice for knowledge base verification in specific domain.

(2) From the point of view of recognition, a piece of knowledge may be right in one time but wrong in another. However, the existing methods mainly aim at the knowledge as a whole. So the anomalies in the globe sense may not be real anomalies in a given time slot. On the contrary, a theory model may not be a real model in application. For example, suppose $\{A \to B,\ B \to C,\ C \to \neg A\}$ is a set of knowledge (rules). Obvious, a model for the set of rules is $A = 0$, $B = 1$, $C = 1$. However, we have commonly the consequence $A \to \neg A$. In real problem, this consequence is unacceptable.

Considering the effect of time in knowledge representation and inference, this paper combines the rule-map technique and state-based knowledge representation technique to realized information verification.

3. Rule-Map based Data Inconsistency Verification

The concept of rule-map is mainly based on the following three points:

1) Formal rules is a kind of important and efficient way to represent human knowledge [14].
2) Knowledge always involves in formal concepts and has a certain hierarchical structure [3].
3) Applying different rules can be seen as recognizing an object from multiple views or aspects.

A. Data rules and rule-map

Without loss of generality, let X be the set of all possible observations for a situation, T be the set of all time slots, $O(t)$ and $E(t)$ be the real observation and the expected observation at time t ($t \in T$) respectively. In the following suppose t is an integer.

***Definition** 3.1:* A potential rule r is a triple $(m(r), g, f)$, where $m(r)$ is an integer, $g : T \to T^{m(r)}$ and $f : X^{m(r)} \to X$ are two maps such that $g(t) = (t_1, t_2, \ldots, t_{m(r)})$ and $E(t) = f(O(t_1), O(t_2), \cdots, O(t_{m(r)}))$, $t_i < t$, $i = 1, 2, \ldots, m(r)$

For example, suppose $H = \{0, 1, 1, 2, 3, 5, 8, \ldots\}$ is a partition of the collected data. Let g and f be $g(t) = (t-1, t-2)$ and $f(O(t_1), O(t_2)) = O(t_1) + O(t_2)$ respectively, then we obtain a potential rule r such that the expected observation of rule r at time slot t is $E(t) = O(t-1) + O(t-2)$.

For convenience, we use $r(t)$ rather than $E(t)$ in the following. Let $t_\perp$ and t^* be the starting and ending time slots for extracting a rule, $T = \{t | t^* \geq t \geq t_\perp\}$, and $T(r)$ be the set of all time slots at which rule r is valid.

***Definition** 3.2:* The feasible degree of a rule r is defined as

$$\delta(r) = \frac{|T(r)|}{|T|}. \tag{1}$$

To a rule r, an observation $O(t)$ is called consistent if $d(r(t), O(t)) < \varepsilon$ where $d(x, y)$ is a given distance between x and y, and ε is an consistent error scale. Let $T(r, \varepsilon) = \{t | d(r(t), O(t)) < \varepsilon\}$.

***Definition** 3.3:* The reliability of a rule r under a given consistent error scale ε is defined as

$$\tau(r, \varepsilon) = \frac{|T(r, \varepsilon)|}{|T(r)|}. \tag{2}$$

The feasible degree of a rule indicates whether a rule can be applied to a given period. The higher feasible degree of a rule, the more applicable it is. However, the feasible degree of a rule cannot guarantee the acceptability of the estimation of it. The reliability of a rule indicates the feasible degree of a rule under the given consistent error scale. As shown in Fig. 1, $t_\perp = 0$, $t^* = 11$, and the extracted rule has estimations at $t = 1, 3, 6, 8, 9, 11$. Hence, the feasible degree of the rule is 0.5. But the reliability of it is 0.83 because five estimations are close to the real observations under given consistent error scale in all six estimations of it.

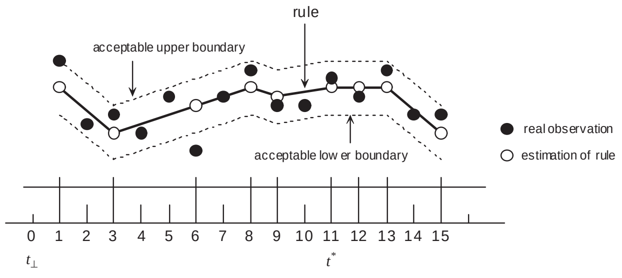

Fig. 1: A rule and its estimations.

Definition 3.4: Two rules r_1 and r_2 are called equivalent if $T(r_1) = T(r_2)$, and $r_1(t) = r_2(t)$ for any $t \in= T(r_1)$ $(= T(r_2))$.

Definition 3.5: Rule r_1 is called to cover rule r_2 if $T(r_1) \supset T(r_2)$ and $r_1(t) = r_2(t)$ for any $t \in T^*$, where $T^* = T(r_2)$.

According to the covering relationship between two rules, a directed graph can be built such that the vertex of it are potential rules, and a pair of nodes with the covering relationship is connected by an arrow as an edge. As shown in Fig. 2, rule r_{11} covers rule r_{21}, but is incomparable with rule r_{12} and rule r_{2m_2}. In the following, we shall call such a directed graph as a rule-map.

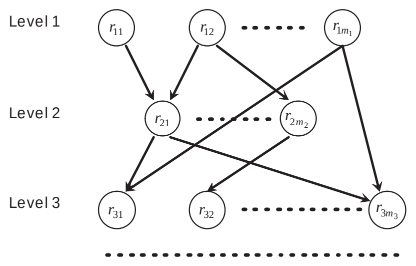

Fig. 2: A rule-map.

A rule-map is used to illustrate the hierarchical structure of human knowledge, which reflects the knowledge before the current time slot. As shown in Fig. 1, the rule is extracted before time slot $t = 11$. Although the rule can be applied to $t = 12, 13, 14, 15$, it doesn't mean the rule will be work at all time slots afterwards. Hence, the picture is updatable.

Suppose G is a rule-map, three features can be concluded:

- the rules at the same level are incomparable to each other,
- the rules at the higher level can take place those at the lower levels in the sense of covering relationship, and
- the rules at the higher level with higher reliability than those at the lower levels.

The first feature indicates that the incomparable rules can describe one object from multiple aspects or viewpoints. The second feature indicates that only a part of rules are necessary to be selected, which are located at the top level of the rule-map. The third feature indicates that each expected observation has higher accuracy for being the conclusion of a rule at the top level. Using these three features of a rule-map, we can efficiently verify data inconsistency.

B. Data Inconsistency Verification

The verification includes four main steps: (1) Construct a rule-map: extracting rules from collected data and constructing a rule-map by the covering relationships. (2) Select rules: selecting the rules which are applicable at the current time slot from the top-level of the rule-map. (3) Estimate a judgment standard: estimating an expected observation of the current time slot as the judgment standard by the selected rules. (4) Check inconsistency: judging if the newly collected data is consistent or not.

More details of the four steps are given as follows.

1) Construct a rule-map: The task includes two main sub-steps. One is to extract rules from collected data by using methods proposed in machine learning, database processing, or time series analysis areas[4], [6]. The other is to establish the covering relationship between any pair of rules if there exists. We take the following strategies to establish the covering relationship between rules (Here the rule is not only limited to the form defined in Definition 3.1 because in general a rule always involves many attributes of a situation.).

Strategy 1 Assume that $r : A \rightarrow B$ and $r' : A' \rightarrow B$ are two extracted rules, where A' is a sub-concept of A, we say that rule r covers rule r'.

Strategy 2 Assume that $r : A \rightarrow B$ and $r' : A' \rightarrow B$ are two extracted rules, where B' is a sub-concept of B, we say that rule r covers rule r'.

Strategy 3 Assume that $r : A \rightarrow B$ and $r' : A' \rightarrow B$ are two extracted rules. If neither rule r covers r' nor r' covers r, then these two rules are incomparable.

Strategy 4 Suppose the extracted rules have been marked in a directed graph. We adjust the rule-map such that for a top-level rule, no rule covers it, and for a non-top-level rule, at least one rule at the higher level covers it.

2) Select rules: By the features of a rule-map, the selected rules must be at the top-level of the rule-map. Moreover, notice that the rules at the top-level may not work at current time slot, we need to select the rules which are applicable at current time slot.

3) Estimate a judgment standard: Without loss of generality, suppose the selected m rules are $r_1, r_2, \cdots, r_m$, and the obtained m estimations are $v_1, v_2, \cdots, v_m$. Then the judgment standard is obtained by the following equation

$$\bar{v} = Agg(v_1, v_2, \cdots, v_m), \tag{3}$$

where Agg is a selected aggregation operator [1] according to the situation at hand. Here, for simplicity, we can select the arithmetical mean of these estimations as:

$$\bar{v} = \frac{\sum_{i=1}^{m} v_i}{m}. \tag{4}$$

Moreover, we can have other choices for aggregation operator, such as the weighted sum, the OWA operator [16], etc.

4) Check the inconsistency: It is common that people are easily to accept one result if it is similar to an expected one. By this idea, the main task at this step is to select an appropriate measure of the closeness between the new observation and the estimated judgment standard. If the closeness between them satisfies a given consistent error scale, then the new observation is consistent data. For convenience, let v be the new observation and $d(x, y)$ be a given distance measurement as closeness measurement. If $d(v, \bar{v}) < \varepsilon$, where ε is a given consistent error scale, then the new observation is acceptable, i.e., it is consistent. Otherwise, it is inconsistent.

Through the four steps, data inconsistency for a set of data can be checked.

4. State-based Logical Inconsistency Verification

A. Knowledge as set of objects' states

For convenience, we shall still use $(x_1, x_2, \dots, x_n)$ to denote a n-tuple only without concerning the order of these elements, and set

$$X_1 \otimes X_2 \otimes \cdots \otimes X_n = \{(x_1, x_2, \dots, x_n) \mid x_i \in X_i, \\ i = 1, 2, \dots, n\}.$$

Suppose T is the set of all time slots, O_1, O_2, …, O_n is a set of objects related to the situation, each object O_i has some possible states denoted by $S_i = \{s_1^{(i)}, s_2^{(i)}, \dots, s_{i_m}^{(i)}\}$, and $S_i(t) \subseteq S_i$ the possible states of object O_i at time t, $i = 1, 2, \dots, n$.

Definition 4.1: A piece of knowledge $\omega(t)$ at time t is defined as follows:

$$\omega(t) \subseteq \bigotimes_{j \in J(\omega)} S_j(t), \tag{5}$$

where $J(\omega) \subseteq \{1, 2, \dots, n\}$ is a set of indexes such that O_j, $j \in J(\omega)$, is an object corresponding to the knowledge $\omega(t)$.

Definition 4.2: A piece of knowledge $\omega(t)$ at time t is called an empty knowledge if for any $j \in J(\omega)$, $S_j^{(\omega)} = \emptyset$ or $S_j^{(\omega)} \subseteq S_j \setminus S_j(t)$, where $S_j^{(\omega)}$ is the set of states of object O_j in knowledge $\omega(t)$.

Empty knowledge represent the impossible states at time t. Obviously, empty knowledge is not unique. Because it is not harmful for our discussion to treat all empty knowledge as one, empty knowledge is denoted by $\mho$ in the following.

Definition 4.3: Let $\omega(t)$ be a piece of knowledge at time t, $\bar{J} = \{i_1, \dots, i_q\} \subseteq J(\omega)$. Then the $\bar{J}$-part of $\omega(t)$ is defined by $\omega(t)|_{\bar{J}}$:

1) $\omega(t)|_{\bar{J}} \subseteq \bigotimes_{j \in \bar{J}} S_j^{(\omega)}$; and
2) for any $(s_{k_1}^{(i_1)}, \dots, s_{k_q}^{(i_q)}) \in \omega(t)|_{\bar{J}}$, there exists $(s_{k_1}^{(i_1)}, \dots, s_{k_q}^{(i_q)}, s_{k_{l_1}}^{(l_1)}, \dots, s_{k_{l_p}}^{(l_p)}) \in \omega(t)$, where $l_1, \dots, l_p \in J(\omega) \setminus \bar{J}$.

In the following, let $\Omega(t)$ be the set of all knowledge at time t. Then the knowledge about a situation is denoted by

$$\Omega = \bigcup_{t \le t_c} \Omega(t), \tag{6}$$

where t_c is the current time. From Eq. (6), we know Ω is incomplete in most cases because it consists of knowledge we obtained before the current time. Also, there are both consistent knowledge and inconsistent knowledge in Ω because a set of knowledge is consistent at one time may not be consistent at another. Although Ω may not be consistent, it is still rational to require $\Omega(t)$ is consistent.

B. Knowledge consistency

Although all knowledge as a whole about a situation may not be consistent, it is still rational to expect the knowledge $\Omega(t)$ at each time slot is consistent.

Definition 4.4: Suppose $\omega(t)$ and $\phi(t)$ are two pieces of knowledge. $\omega(t)$ and $\phi(t)$ are said to be strict consistent if

1) $J(\omega) \cap J(\phi) = \emptyset$; or
2) $J(\omega) \cap J(\phi) \neq \emptyset$ and $S_j^{(\omega)} = S_j^{(\phi)}$ for any $j \in J(\omega) \cap J(\phi)$.

Definition 4.5: Suppose $\omega(t)$ and $\phi(t)$ are two pieces of consistent knowledge. If $J(\omega) \cap J(\phi) \neq \emptyset$ and

$$S_j^{(\omega)} \neq S_j^{(\phi)} \text{ and } S_j^{(\omega)} \cap S_j^{(\phi)} \neq \emptyset \tag{7}$$

for any $j \in J(\omega) \cap J(\phi)$, then $\omega(t)$ and $\phi(t)$ are said to be partial consistent.

Definition 4.6: Suppose $\omega(t)$ and $\phi(t)$ are two pieces of knowledge. If $J(\omega) \cap J(\phi) \neq \emptyset$, and there exists $j \in J(\omega) \cap J(\phi)$ such that

$$S_j^{(\omega)} \cap S_j^{(\phi)} = \emptyset, \tag{8}$$

then $\omega(t)$ and $\phi(t)$ are said to be inconsistent.

The inconsistency of two pieces of knowledge indicates that any object cannot have states in two separated states sets at the same time. This can be seen as an expansion of the inconsistency of the classical logic, where a logical formula cannot take truth values 1 and 0 simultaneously.

Before define the inconsistency of a set of knowledge, we first discuss the combination of two pieces of knowledge, which includes two forms. One is extracting the common part of them, which is denoted by $E(\omega, \phi)$, and the other one is coupling them and cutting the inconsistent components of them, which is denoted by $C(\omega, \phi)$, where $\omega(t)$ and $\phi(t)$ are the combined knowledge.

Definition 4.7: Let $\omega(t)$ and $\phi(t)$ be two pieces of consistent knowledge. Then $E(\omega, \phi)$ is obtained by:

1) When $J(\omega) \cap J(\phi) \neq \emptyset$.
   - $J(E(\omega, \phi)) = J(\omega) \cap J(\phi)$;
   - $E(\omega, \phi) \subseteq \bigotimes_{j \in J(\omega) \cap J(\phi)} S_j^{(\omega)} \cap S_j^{(\phi)}$; and
   - $E(\omega, \phi) \subseteq \omega(t)|_{J(\omega) \cap J(\phi)}$; and
   - $E(\omega, \phi) \subseteq \phi(t)|_{J(\omega) \cap J(\phi)}$.
2) When $J(\omega) \cap J(\phi) = \emptyset$. Define $E(\omega, \phi)$ as $\omega(t) \cup \phi(t)$, where $\cup$ is the ordinary joint of two sets.

Definition 4.8: Let $\omega(t)$ and $\phi(t)$ be two pieces of consistent knowledge. Then $C(\omega, \phi)$ is obtained by:

1) When $J(\omega) \cap J(\phi) \neq \emptyset$.
   - $J(C(\omega, \phi)) = J(\omega) \cup J(\phi)$;
   - $C(\omega, \phi)|_{J(\omega) \cap J(\phi)} = \omega(t)|_{J(\omega) \cap J(\phi)} \cap \phi(t)_{J(\omega) \cap J(\phi)}$;
   - $(C(\omega, \phi)|_{J(\omega)})|_{J(\omega) \cap J(\phi)} = C(\omega, \phi)|_{J(\omega) \cap J(\phi)}$;
   - $(C(\omega, \phi)|_{J(\phi)})|_{J(\omega) \cap J(\phi)} = C(\omega, \phi)|_{J(\omega) \cap J(\phi)}$.
2) When $J(\omega) \cap J(\phi) = \emptyset$. Define $C(\omega, \phi)$ as $\omega(t) \otimes \phi(t)$.

In the following, we also denote $C(\omega, \phi)$ as $\omega \sqcap \phi$ and $E(\omega, \phi)$ as $\omega \sqcup \phi$.

Base on Definition 4.8, let $\Omega^*(t) \subseteq \Omega(t)$, and

$$C(\Omega^*(t)) = \bigsqcap_{\omega(t) \in \Omega^*(t)} \omega(t). \tag{9}$$

We shall call a set Ω of knowledge is indivisible if there doesn't exist a division of $J(\Omega) = J_1 \cup J_1 \cup \cdots \cup J_k$ such

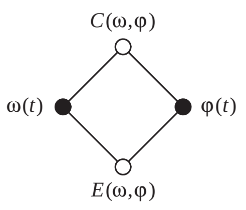

Fig. 3: Covering among knowledge

that for any $\omega \in \Omega$, $J(\omega) \subseteq J_l$, where $l \in \{1, 2, \ldots, k\}$ and $J(\Omega) = \bigcup_{\omega\in\Omega} J(\omega)$.

***Definition** 4.9:* A set of knowledge $\Omega(t)$ is called consistent if $C(\Omega^*(t)) \nsubseteq \mho$ for any indivisible knowledge subset of $\Omega(t)$; otherwise, it is called inconsistent.

C. Knowledge covering

***Definition** 4.10:* Two pieces of knowledge $\omega(t)$ and $\phi(t)$ are said to be equivalent and denoted by $\omega(t) \equiv \phi(t)$ if $J(\omega) = J(\phi)$ and $S_j^{(\omega)} = S_j^{(\phi)}$ for any $j \in J(\omega)(= J(\phi))$.

***Definition** 4.11:* A piece of knowledge $\omega(t)$ is said to be a logical consequence of knowledge $\phi(t)$ and denoted by $\phi(t) \models \omega(t)$ if the following conditions hold:

1) $J(\omega) \subseteq J(\phi)$,
2) $\omega(t) = \phi(t)|_{J(\omega)}$.

Obviously, two equivalent knowledge ω and ϕ are logical consequence of each other. It is easy to verify that the following conclusions hold.

1) $\omega(t) \models \omega(t)$ for any $\omega(t) \in \Omega(t)$.
2) $\omega(t) \equiv \phi(t)$ if $\omega(t) \models \phi(t)$ and $\phi(t) \models \omega(t)$.
3) $\omega(t) \models \psi(t)$ if $\omega(t) \models \phi(t)$ and $\phi(t) \models \psi(t)$.

By Definition 4.11, we know $E(\omega, \phi)$ is a logical consequence of both $\omega(t)$ and $\phi(t)$. So, $\{\omega(t), \phi(t)\} \models E(\omega, \phi)$. Similarly, $C(\omega, \phi) \models \omega(t)$ and $C(\omega, \phi) \models \phi(t)$.

The logical consequence relationship between knowledge gives a hierarchical structure among all knowledge at time t. We draw the hierarchical structure as follows:

$$\omega(t) \preccurlyeq \phi(t) \text{ if and only if } \phi(t) \models \omega(t). \tag{10}$$

The relationship $\preccurlyeq$ is called a covering relationship, which can be pictured in a graph such as Figure 3.

D. Logical Inconsistency Verification

According to the construction of covering relationship, we know the $C(\Omega(t))$ must be at the top level, which exactly includes all possible states for each objects, which satisfy all knowledge in $\Omega(t)$, and can be applied to verify logical inconsistency of observations.

1) Problem illustration: The problem of checking logical inconsistency can be formally described as: In a situation, we have stored some related knowledge $\Omega = \cup_{t\in T}\Omega(t)$ about the situation, which are a set of collections of states of a set of objects $O = \{O_1, \ldots, O_n\}$. At a given time slot t, we collected some new observations $S^* = \{S_j^* \mid j \in J(S^*)\}$ of some objects $O^* = \{O_j^* \mid j \in J(O^*)\} \subseteq O$, where $J(S^*) = J(O^*)$ is an indexes set. Our question is whether these new observations are consistent or not with the stored knowledge from the viewpoint of logical consistency.

2) Basic idea: Assume that the stored knowledge at time t is consistent. Under this assumption, the potential inconsistencies are introduced by the new observations. As all possible combinations of observations which satisfy all knowledge can only fall in $C(\Omega(t))$, hence, that checking logical inconsistency of observations is equal to checking whether it is fall in $C(\Omega(t))$.

3) Processing steps: The presented method includes the following steps.

Step 1: Compare $J(S^*)$ and $J(C(\Omega(t)))$. If $J(C(\Omega(t))) \supseteq J(S^*)$, then go to Step 2. If $J(C(\Omega(t))) \subset J(S^*)$, then go to Step 3.

Step 2: Check whether $S^* \in (C(\Omega(t)))|_{J(S^*)}$ or not. If $S^* \in (C(\Omega(t)))|_{J(S^*)}$, then the observations are consistent; otherwise, they are inconsistent. Go to Step 5.

Step 3: Divide S^* into two parts S_1^* and S_2^* such that $J(S_1^*)) = J(C(\Omega(t)))$. For S_1^*, go to Step 2. For S_2^*, using rule-map method to check their consistencies of observation $s \in S_2^*$. If some observations in S_2^* is inconsistent, then the observations is inconsistent. go to Step 4.

Step 4: Update knowledge base. For any consistent observation $s \in S_2^*$, construct $C(C(\Omega(t)), s)$ and add $C(C(\Omega(t)), s)$ to $\Omega(t)$.

Step 5: Stop.

5. Examples

In this section, we shall give two simple examples to describe the presented methods for information verification.

***Example** 5.1:* Suppose a situation has four states 0, 1, 2, and 3, and has 17 historical data, which are listed in Table 1.

Table 1: A data set

t	$O(t)$	$r_1(t) = O(t-1)$	$r_2(t) = O(t-2)$	$r_3(t) = 2$
1	2	N	N	T
2	2	T	N	T
3	1	F	F	F
4	1	T	F	F
5	2	F	F	T
6	2	T	F	T
7	2	T	T	T
8	3	F	F	F
9	2	F	T	T
10	2	T	F	T
11	3	F	F	F
12	2	F	T	T
13	2	T	F	T
14	2	T	T	T
15	2	T	T	T
16	2	T	T	T
17	2	T	T	T
18	2			
19	2			
20	3			

Suppose the rules r_1, r_2, and r_3 are obtained three rules. Obviously, r_1 covers entries from 2 to 17, while r_2 covers entry 3 to 17, and r_3 covers entry 1 to 17. notice that the three rules are incomparable to each other. So they all belong to the top-level of a mule-map, and will be selected as the rules for estimating the expected observation.

1) Let $t = 18$. Then $r_1(18) = 2$, $r_2(18) = 2$, and $r_3(18) = 2$. Hence by the presented method, the expected observation is the state 2(=(2 + 2 + 2)/3). Because the real observation is the state 2, it satisfies any given error and so is inconsistent.
2) Let $t = 20$. Then $r_1(20) = 2$, $r_2(20) = 2$, and $r_3(20) = 2$. So, the expected observation is state 2 but the real observation is state 3. Hence, the real observation is inconsistent to the expected one.

***Example** 5.2:* Suppose we have a knowledge base $\Omega(t)$ which have the following knowledge shown in Table 2, where ω_i, $i = 1, 2, \ldots, 10$, is a piece of knowledge, and p_j, $j = 1, 2, \ldots, 14$, are related objects and each of them have two possible states 1 and 0. Now, let $S^* = \{$, $(p_1 = 1$, $p_2 = 1$, $p_3 = 1$, $p_9 = 1)\}$. By our method, we have

TABLE 2: A KNOWLEDGE BASE.

No.	Knowledge
ω_1	$\{(p_1 = 1, p_2 = 1, p_5 = 1, p_6 = 1)\}$
ω_2	$\{(p_2 = 1, p_{14} = 1)\}$
ω_3	$\{(p_6 = 1, p_{10} = 1)\}$
ω_4	$\{(p_3 = 1, p_4 = 1, p_7 = 1)\}$
ω_5	$\{(p_{10} = 1, p_{14} = 1)\}$
ω_6	$\{(p_7 = 1, p_{10} = 1, p_{11} = 1)\}$
ω_7	$\{(p_8 = 1, p_{11} = 1)\}$
ω_8	$\{(p_8 = 1, p_7 = 1)\}$
ω_9	$\{(p_{10} = 1, p_{12} = 1)\}$
ω_{10}	$\{(p_{10} = 1, p_{13} = 1)\}$

Step 1: $J(S^*) \nsubseteq J(C(\Omega(t)))$. Then goto Step 3.

Step 3: Dividing S^* into $S_1^* = \{(p_1 = 1, p_2 = 1, p_3 = 1)\}$ and $S_2^* = \{(p_9 = 1)\}$. For S_1^*, goto Step 2. For S_2^*, without loss of generality, suppose it is consistent by the rule-map method. Then goto Step 4.

Step 2: For S_1^*, we know it is consistent.

Step 4: Because the S^* is a set of consistent observation, we shall update our knowledge by coupling $\{(p_9 = 1)\}$ and $C(C(\Omega(t)))$ and have $\{(p_1 = 1$, $p_2 = 1$, $p_3 = 1$, $p_4 = 1$, $p_5 = 1$, $p_6 = 1$, $p_7 = 1$, $p_8 = 1$, $p_9 = 1$, $p_{10} = 1$, $p_{11} = 1$, $p_{12} = 1$, $p_{13} = 1$, $p_{14} = 1)\}$.

Step 5. Stop.

6. CONCLUSION

In this paper, we proposed a rule-map technique and a state-based knowledge representation method to verify data inconsistency and logical inconsistency respectively. According to the given examples and related experiments, the following topics will be our future research goals: (1) how to combine these two techniques, (2) how to efficiently construct a rule-map, and (3) how to representation domain knowledge by the state-based representation technique.

ACKNOWLEDGMENTS

The work presented is supported by Australia Research Council (ARC) under discovery grant DP0559213.

REFERENCES

[1] G. Beliakov and J. Warren, "Appropriate choice of aggregation operators in fuzzy decision support systems", *IEEE Trans. on Fuzzy Systems*, vol. 9, no. 6, pp. 773–784, 2001.

[2] N. Botten, "Complex knowledge-base verification using matrices", *Lecture Notes in Artificial Intelligence*, vol. 604, pp. 225–235, 1992.

[3] D. Genest and M. Chein, "A content-search information retrieval process based on conceptual graphs", *Knowledge and Information Systems*, vol. 8, pp. 292–309, 2005.

[4] T. P. Hong and C. Y. Lee, "Induction of fuzzy rules and membership functions from training examples", *Fuzzy Sets and Systems*, vol. 84, pp. 33–47, 1996.

[5] W. Fritz (1997), "Intelligent Systems and Their Societies", free e-book [Online]. Available: http://www.intelligent-systems.com.ar/intsyst/index.html

[6] F. Höppner, "Discovery of Temporal Patterns: Learning Rules about the Qualitative Behaviour of Time Series", in *Proc. 5th European Conference on Principle and Practice of Knowledge Discovery in Databases*, Freiburg, Germany, 2001, pp. 192–203.

[7] B. Mazure, L. Saïs and Eric Grégoire, "Checking several forms of consistency in nonmonotonic knowledge-bases", in *Qualitative and Quantitative Practical Reasoning, First International Joint Conference on Qualitative and Quantitaitve Practical Reasoning ECSQARU-FAPR'97*, Bad Honnef, Germany, *Lecture Notes in Computer Sciences*, vol. 1244, pp. 122–130, 1997.

[8] C. Mues and J. Vanthienen, "Efficient rule base verification using binary decision diagrams", in *Proc. of Database and Expert Systems Application 2004*, *Lecture Notes in Computer Science*, vol. 3180, pp. 445–454, 2004.

[9] J. H. Park and P. H. Seong, "An integrated knowledge base development tool for knowledge acquisition and verification for NPP dynamic alarm processing systems", *Annals of Nuclear Energy*, vol. 29, no. 4, pp. 447–463, 2002.

[10] Z. Pawlak and A. Skowron, "Rough sets and Boolean reasoning", *Information Sciences*, 2006. (in press)

[11] Z. Pawlak and A. Skowron, "Rudiments of rough sets", *Information Sciences*, 2006. (in press)

[12] F. Polat, "UVT: A unification-based toold for knowledge base verification", *IEEE Expert–Intelligent Systems & Their Applications*, vol. 8, no. 3, pp. 69–75, 1993.

[13] A. D. Preece and R. Shinghal, "Foundation and application of knowledge-base verification", *International Journal of Intelligent Systems*, vol. 9, no. 8. pp. 683–701, Aug. 1994.

[14] S. J. Russell and P. Norvig, "Artificial Intelligence: A Modern Approach", 2nd Edition, Prentice Hall, NJ, 1995.

[15] P. Wu and S. Y. W. Su, "Rule validation based on logical deduction", in *Proc. 2nd International Conference on Information and Knowledge Management*, pp. 164–173, 1993.

[16] R. R. Yager, "OWA aggregation over a continuous interval argument with applications to decision making", *IEEE Trans. Systems, Man, and Cybernetics—Part B: Cybernetics*, vol. 34, no. 5, pp. 1952–1963, 2004.

[17] S. J. H. Yang, J. J. P. Tsai and C. C. Chen, "Fuzzy rule base systems verification using high-level Petri Nets", *IEEE Transactions on Knowledge and Data Englineering*, vol. 15, no. 2, pp. 457–473, 2003.

[18] D. Zhang and Luqi, "Approximate declarative semantics for rule base anomalies", *Knowledge-Based Systems*, vol. 12, pp. 341–353, 1999.

[19] J. Lu and G. Zhang, "Information integration based team situation assessment in an uncertain environment", in *Proc. of FLINS2006*, Genoa, Italy, in D. Ruan, et al. *Applied Artificial Intelligence*, World Scientific Press, 2006, pp. 441–448.