Proceedings of the Korean Nuclear Society Autumn Meeting
Seoul, Korea, October 1995

A Knowledge - Base Verification of NPP Expert systems using Extended Petri Nets

Il Won Kwon and Poong Hyun Seong
Korea Advanced Institute of Science and Technology

Abstract

The verification phase of knowledge base is an important part for developing reliable expert systems, especially in nuclear industry. Although several strategies or tools have been developed to perform potential error checking, they often neglect the reliability of verification methods. Because a Petri net provides a uniform mathematical formalization of knowledge base, it has been employed for knowledge base verification. In this work, we devise and suggest an automated tool, called COKEP (Checker Of Knowledge base using Extended Petri net), for detecting incorrectness, inconsistency, and incompleteness in a knowledge base. The scope of the verification problem is expanded to chained errors, unlike previous studies that assumed error incidence to be limited to rule pairs only. In addition, we consider certainty factor in checking, because most of knowledge bases have certainty factors.

1. Introduction

The adoption of expert systems mainly as operator support systems is becoming gradually popular in nuclear industry as the control algorithms of nuclear power plant system become more and more sophisticated and complicated. As a result of this popularity, a large number of expert systems are developed, and most of these systems employ a rule-based formalism for knowledge representation since it is the simplest knowledge representation method to develop. In spite of this advantage, incorrectness, inconsistency, and incompleteness may be inadvertently brought into the knowledge base because it is often built in an incremental process. In other words, such anomalies may occur at any stage in the knowledge transfer process that is to transfer expertise from the human expert into the computer by the knowledge engineers. Therefore, it is widely noted that assuring the reliability of knowledge-based systems is very important, especially in

nuclear industry, and it is also recognized that the process of verification is an essential part of reliability assurance for these systems.

The traditional approaches to knowledge base verification, which have generally involved each rule comparison and decision context enumeration[1], are computationally expensive. They used domain-specific information in the verification process. Verification tools and algorithms developed later, such as COVADIS[2] and EVA[3], were based on a variety of approaches and were capable of detecting more subtle cases of anomalies. In recent days, a Petri net-based verification method was proposed[4]. A numerical Petri net was used to model knowledge base of production rules, and reachability analysis was then conducted to reveal inconsistency and incompleteness in a knowledge base[5]. In PREPARE[6], anomalies in a knowledge base are defined in terms of the Pr/T net model. Then, these terms are identified by using syntactic pattern recognition method.

The conventionally proposed methods are inefficient for large systems with chained errors and certainty factor errors. In order to extend previous researches we consider verification of knowledge base having certainty factor, in global level as well as in local level. We devise and suggest an automated tool, called COKEP (Checker Of Knowledge base using Extended Petri net), for detecting incorrectness (redundant, subsumed, circular rules), inconsistency (conflict rules), and incompleteness (unreachable conclusion, unreferenced conditions, isolated, omitted rules) in a knowledge base.

2. Anomalies in Knowledge base

The anomalies in knowledge base can be divided into three types, that is incorrectness, inconsistency, and incompleteness. The illustration of them is following:

Incorrectness

Redundant rules: Two rules are redundant if they contain the same conditions and have the same conclusions. Since the rules might count twice, the weight of their conclusions is increased.

Subsumed rules: One rule is subsumed by another if the two rules have the same conclusions, but one contains additional constraints on the conditions. The rule *IF (a(x) and a(y)) THEN c(z)* is subsumed by the rule *IF a(x) THEN c(z)*, where x, y, and z are variables, and a and c are logical relationships. Whenever the more restrictive rule succeeds, the less restrictive rule also succeeds, resulting in redundancy. Additionally, the more restrictive rules add weight to the conclusions by the less restrictive rules.

Circular rules: A set of rules is circular if the chaining of rules in a set forms a cycle. If we have the set of rules such as *IF a(x) THEN b(x)*, *IF b(x) THEN c(x)*, and *IF c(x) THEN a(x)*, the systems enter infinite loop at run time unless the system has a special way of handling circular rules. Also, this definition includes the possibility of a single rule to form a cycle, i.e., *IF a(x) THEN a(x)*.

Inconsistency

Conflicting rules: Two rules are conflicting if they have the same conditions but with conflicting conclusions. Formally, the rule *IF a(x) THEN b(y)* is contradictory to the rule *IF a(x) THEN c(y)*. Sometimes, given the same set of symptoms, the expert might wish to conclude different conclusions with different certainty factors.

Incompleteness

Unreachable conclusion rules: In a goal driven production systems, the conclusion of a rule should match either a goal or an IF condition of another rule. If there are no matches, it is unreachable. If a chain of rules produces a certainty factor less than the threshold, it is also unreachable.

Unreferenced condition rules: If the one of rule conditions does not appear as the conclusion of another rule and as a known fact, it is an unreferenced condition. Rules with an unreferenced condition may indicate the possibility of some other rules or facts missing.

Isolated rules: If all of conditions are an unreferenced condition and the conclusion is an unreachable conclusion, then the rule is called as an isolated rule. The presence of an isolated rule may indicate the possibility of missing rules.

Missing rules: The knowledge base has deficiency when the rule will not produce any output. These rules can be indicated by unreachable conclusion rules, unreferenced conditions rules, and isolated rules.

3. Knowledge Base Verification Tool

The structure of a extended Petri net is defined by its places, transitions, input functions, and output functions

3.1 Extended Petri nets

An extended Petri net is composed of six parts: a set of place P, a set of transition state place P', a set of transitions T, input functions I_1, input functions I_2, and output functions O. The input and output functions are related to transitions and places. $P = \{p_1, p_2, \ldots, p_n\}$ is a finite set of places, $n \geq 0$. $P' = \{p'_1, p'_2, \ldots, p'_m\}$ is a finite set of transition state place, $m \geq 0$. $T = \{t_1, t_2, \ldots, t_m\}$ is a finite

transitions, $m \geq 0$.

Input places of the transition, in the extended Petri net, are classified into two types. 'A' is the output place of a different rule, which is used for searching the path of chained rules. 'B' is the initial marking place, which is used for finding the known fact of chained rules(Fig.1). Input functions I_1 and I_2 are made by each input place, 'A' and 'B'. Place 'C' is the transition state place that informs whether transition is fired or not. Since the places 'A' and 'B' maintain the information of known fact, after firing transition, another place,'C', is required to check the firing transition.

3.2 Matrix Equations

An anomaly detection approach is based on a matrix view of Petri nets. Three matrices D^-_1, D^-_2, and D^+ to represent the input and output functions can be defined from the (P, P', T, I_1, I_2, O) definition of extended Petri nets. Each matrix has m rows for each transition and n columns for each place. $e[j]$ means the unit m-vector that is zero everywhere except in the jth component. The transition t_j is represented by the unit m-vector $e[j]$.

A transition t_j is enabled in a symbol μ if $\mu \geq e[j] \cdot (D^-_1 + D^-_2)$. The result vector, defined as $\delta(\mu, t_j)$, of firing transition t_j in a marking μ is

$$\begin{aligned} \delta(\mu, t_j) &= \mu - (e[j] \cdot (D^-_1 + D^-_2)) + (e[j] \cdot D^+) \\ &= \mu + e[j] \cdot (D^+ - (D^-_1 + D^-_2)) \\ &= \mu + e[j] \cdot D \end{aligned} \tag{1}$$

where,

$D = D^+ - (D^- + D^-)$, $D^-_1[j,i] = \#(p_i, I_1(t_j))$, $D^-_2[j,i] = \#(p_i, I_2(t_j))$, and $D^+[j,i] = \#(p_i, O(t_j))$.

Now for a sequence of transition firings $\sigma = t_{j1}, t_{j2}, \ldots, t_{jk}$,

$$\begin{aligned} \delta(\mu, \sigma) &= \delta(\mu, t_{j1}, t_{j2}, \ldots, t_{jk}) \\ &= \mu + (e[j_1] \cdot D) + (e[j_2] \cdot D) + \cdots + (e[j_k] \cdot D) \\ &= \mu + (e[j_1] + e[j_2] + \cdots + e[j_k]) \cdot D \\ &= \mu + f(\sigma) \cdot D \end{aligned} \tag{2}$$

The vector $f(\sigma) = e[j_1] + e[j_2] + \cdots + e[j_k]$ is called the *firing vector* of the sequence $t_{j1}, t_{j2}, \ldots, t_{jk}$.

4. Anomaly Detection

An alternative strategy is provided in this work, transforming the problem of verification into that of reachability of specific states in the net. As the detection of anomalies is based on the

results of firing transition, verification problems can be expressed as reachability problems. In order to solve these problems matrix analysis of the extended Petri net and backward chaining methods of the rule set are employed.

The matrix analysis has some problems in checking anomalies. The result vector of firing transition t_j in marking μ is a necessary but not sufficient condition for reachability analysis in chained rule set. A backward chaining method in the rule set is used for solving this problem. First, the result vector of firing transition is obtained by the matrix analysis, eq.(1) and (2). Then, we find chained rule path using matrix D^-_1 and backward chaining method. The conditions for chained rule transition can be acquired by matrix D^-_2 which has the information of initial marking places. The certainty factor checking is performed after finding the chained rule path. The general procedure of these checking process is shown in (Fig. 2).

5. Conclusion

The verification has, until now, focused upon building novel anomaly detection systems and improving the efficiency of existing systems. The issues of theoretical foundations of knowledge base verification, however, have remained unaddressed. This work provides reliable verification method that adopts improved verification techniques and an automated integral verification tool. COKEP is based on modeling a knowledge base by using the extended Petri net, and uses matrix analysis and backward chaining methods in verification process.

The scope of the verification is expanded to chained errors, unlike previous studies that assumed error incidence to be limited to rule pairs only. In addition, we consider certainty factor in checking, because most expert systems have certainty factors.

References

1. B.J. Cragun and H.J. Studel, "A decision-table-based processor for checking completeness and consistency in rule-based expert systems," Int. J. Man-Machine Studies, vol. 26, pp. 633-648, 1987.
2. M.R. Rousset, "On the consistency of knowledge bases: COVADIS system," Proc.8th Eur. Conf. AI, pp. 79-84, 1988.
3. C.L.Chang, J.B.Combs, and R.A.Stachowitz, "A report on the expert systems validation associate(EVA)," Expert Systems with Applications, Vol. 1, pp. 217-230, 1990.
4. P.Meseguer, "A new method to checking rule bases for inconsistency: A Petri net approach," Proc. 9th Eur. Conf. AI, pp.437-442, 1990.
5. N. K. Liu and T. Dillon, "An approach towards the verification of expert systems using numerical petri nets," Int. J. Intell. Sys., vol. 6, pp. 255-276, 1991.
6. D. Zhang and D. Nguyen, "A tool for knowledge base verification," IEEE Trans. Knowl. Data Eng., vol.6, no. 6, pp. 983-989, Dec. 1994.

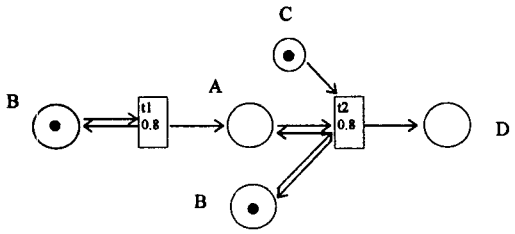

Fig. 1 A example of the extended Petri net model

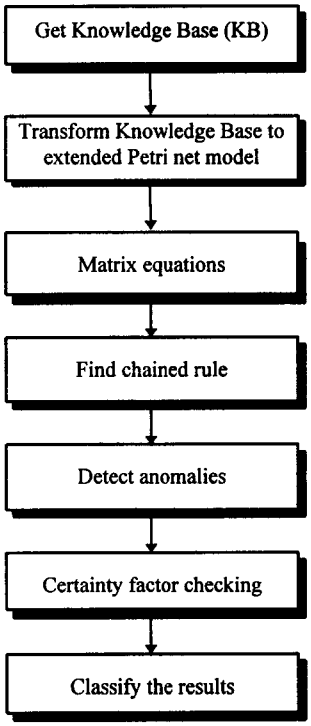

Fig. 2 The schematic diagram of the anomalies detection procedure