Formal Verification of Some Potential Contradictions in Hybrid Expert Systems

Simon C.K. Shiu, James N.K. Liu, Daniel S. Yeung

Department of Computing
Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong
E-mail: {csckshiu | csnkliu | csdaniel}@comp.polyu.edu.hk

ABSTRACT

With increasingly complex, sophisticated and changing real-world situations, it has been recognized that expert systems which combine one or more techniques greatly increase the problem solving capability and help overcome some of the shortcomings associated with any single technique. The verification of these expert systems requires methods that could tackle the multiple knowledge representation paradigms and integrated inference mechanisms used. This paper describes a formal technique based on State Controlled Coloured Petri Nets (SCCPNs) for verifying some potential contradictions in Hybrid Expert Systems (HES) that emphasize an integration of object hierarchy, property inheritance and production rules. The main idea of our approach is to convert the HES into a State Controlled Coloured Petri Net where the object hierarchy, property inheritance and production rules are modelled as separated components in the same SCCPN. The detection and analysis of the potential contradictions in the system are done by constructing and examining the reachability tree spanned by the knowledge inference. Propositions are formulated to verify such potential contradictions, and their mathematical proofs are explained.

1. INTRODUCTION

Traditionally, attention has been concentrated on using verification techniques to tackle rule-based systems [8], [9], [13], [14]. However, these techniques exhibit a limited range of applicability. They could not cope with the kind of Hybrid Expert Systems (HES), e.g. Rule-based plus Frame-based, which many of the current Expert Systems are being developed [2], [5], [16], [22]. The use of this hybrid approach integrates the power of organizing data objects in a class hierarchy and reasoning about the objects through user pre-defined logical associations. This advantage accounts for many popular Expert System development software (or shells), such as ADS, ART, EXSYS EL, KAPPA-PC, KBMS, NEXPERT OBJECT, LEVEL5 OBJECT, PRO-KAPPA, REMIND, which combine some sort of Frame-based representation with a Rule-based inference engine.

The verification of these Hybrid Expert Systems requires methods that could tackle the multiple knowledge representation paradigms and integrated inference mechanisms used. This paper presents a formal description technique based on State Controlled Coloured Petri Nets for verifying some potential contradictions in Hybrid Expert Systems that emphasizes an integration of object hierarchy, property inheritance and production rules.

2. A HYBRID EXPERT SYSTEM (HES)

A Hybrid Expert System combines multiple representation paradigms into a single integrated environment. For a Rule- and Frame-based integration, it composes of the following key features: Object Classes, Slot Attributes, Inheritance Relations, Demons, Methods, Rules and Reasoning Strategies. These features can be analyzed using three conceptual views [6] of an Expert System, they are: (1) An Object View which encapsulates a module of knowledge (or a concept). These knowledge modules (concepts) are represented by Object Classes. Inheritance Relations describe how these knowledge modules are related. (2) A Function View which specifies the functional behaviour of the objects within the Expert System. These functions are represented using Methods and Demons. (3) A Control View which specifies the sequence of knowledge inference in the Expert System. These controls are represented in terms of Rules and Reasoning Strategies.

In practical HES development [18], [19], Frames are used to represent domain objects, various kinds of Demons are used to implement procedures attached to specific slots, Inheritance is used to inherit Class properties, Methods and Demons among Object Classes, Message Passing is used for the interaction among different objects and Methods are used to perform algorithmic actions or some array manipulation within an object. Rules are used to describe heuristic problem-solving knowledge, Forward and Backward chains are commonly used to reason using rules. Therefore, in HES, the Frame base can be seen as being used to define the vocabulary for the Rule base, i.e. the possible values that slots can be defined and so specified, and the literal used to construct rules must conform to the restrictions imposed by what is available from the class hierarchy. The Frame base is married together with the Rules designed to manipulate it. The specific integration mechanisms of HES are as follows:

- Rules with Message Passing: Rules send or receive messages to and from objects for testing the Rules' premises.
- Rules with Inheritance: Rules directly read and write data into slots in a parent object and through inheritance of this slot's value to its children objects, trigger other rules to fire.
- Rules with Demons: Rules directly read and write data into slots and cause the execution of the associated Demons, which then trigger other rules to fire.
- Rules with Methods: Rules are embedded as part of an object's methods. Since methods are arbitrary pieces of code attached to an object, they can access the rules through function calls.

- Rules with Instances: Rules can be used to create/delete an instance of a specific Object Class.

- Backward Chain with Inheritance: Goal directed search with inheritance as one of the means to establish the rule chains linking up different Object Classes.

- Forward Chain with Inheritance: Data directed search with inheritance as one of the means to establish the rule chains linking up different Object Classes.

3. SCCPN REPRESENTATION OF HES

As shown in Table 3.1 the components of the HES are separately represented, which can be modelled explicitly by the SCCPN. The places are taken to correspond to predicates and object classes, and transitions to represent rules implications as well as inheritance. There are two major types of tokens, one is the state token which records the state of the predicate and the class type information. (i.e. Since rules may be fired by either parent class instance or child class instances). The second type of token is the object instance token which represents a particular object instance of a particular class within the object hierarchy. Transitions are fired to represent rules being executed or inheritance is being carried out. The maximum number a rule can be executed is equal to the total number of different class types. (i.e. each class type object instance can fire a particular rule once at most). Each input place of a rule has a self-loop arc for maintaining the state of the predicate. Similarly, the input place of an inheritance also has a self-loop arc for recording the inheritance execution. Methods and Demons are represented by functions in the arc inscription of the SCCPN. The net result is the exchange of colour tokens from places to places and a new marking, which is defined as the distribution of tokens over the places of the SCCPN, is obtained.

Hybrid Expert System	State Controlled Coloured Petri Net
Frame-based part	
Object Classes	Places
Object Class Types	Colour Sets
Object Instances	Tokens
Slots	Variables in Tokens
Facts in Slots	Binding of Variables with Constants
Inheritances	Transitions
Demon	Arc Expressions
Methods	Arc Expressions
Rule-based part	
Predicates	Places
Predicates States	Tokens
Rules	Transitions
Facts	Binding of Variables with Constants
Transition Operations	Arc Expressions

Table 3.1. Conceptual interpretation of HES in SCCPNs.

The SCCPN notation employed is an extension of State Controlled Petri Nets proposed by [13], and Coloured Petri Nets proposed by [10], [11], and is specified as follows.

DEFINITION 3.1. A SCCPN can be defined as a 10-tuple given by = (Σ, *P, T, D, F, A, N, C, E, I*), where satisfying the requirements below:

$\Sigma = \{ \omega_1, \omega_2, \ldots, \omega_i \}$, a finite set of non-empty types, called *colour sets*, $i \geq 1$,

$P = \{P_c, P_r\}$ a finite set of places,

$P_c = \{ p_{c1}, p_{c2}, \ldots, p_{cj} \}$, a finite set of places that model the classes of the HES, called class places, $j \geq 1$,

$P_r = \{ p_{r1}, p_{r2}, \ldots, p_{rk} \}$, a finite set of places that model the predicates of the production rules, called predicate places, $k \geq 1$,

$P_c \cap P_r$: the intersection of $P_c \cap P_r$ represents those IS-A predicates of the rule sets attached to the specific classes,

$T = \{ T_c, T_r \}$, a finite set of transitions,

$T_c = \{ t_{c1}, t_{c2}, \ldots, t_{cl} \}$, a finite set of transitions that are connected to and from class places, called inheritance transition, $l \geq 1$,

$T_r = \{ t_{r1}, t_{r2}, \ldots, t_{rm} \}$, a finite set of transitions that are connected to or from predicate places, called predicate transition, $m \geq 1$,

$T_c \cap T_r = \emptyset$,

$D = \{ d_1, d_2, \ldots, d_n \}$, a finite set of predicates, $|P_r| = |D|$, $n \geq 1$,

$F = \{ f_1, f_2, \ldots, f_n \}$, a finite set of classes, $|P_c| = |F|$, $n \geq 1$,

$A = \{ a_1, a_2, \ldots, a_k \}$, a finite set of arcs, $k \geq 1$, $P \cap T = P \cap A = T \cap A = \emptyset$,

$N : A \rightarrow P \times T \cup T \times P$, a node function, it maps each arc into a pair where the first element is the source node and the second is the destination node, the two nodes have to be of different kinds. The node functions can be further classified into the following eight different types:

Inheritance : $\{ \tilde{A}_c, \ddot{A}_c, \tilde{A}_s, \ddot{A}_s \}$ where

$\tilde{A}_c : T_c \rightarrow (P_c)_{MS}$ is an input class function for inheritance, a mapping from inheritance transitions to the bags of class places. The word $_{MS}$ stands for multi-set (or bags).

$\ddot{A}_c : T_c \rightarrow (P_c)_{MS}$ is an output class function for inheritance, a mapping from inheritance transitions to the bags of class places.

$\tilde{A}_s : T_c \rightarrow (P_c)_{MS}$ is an input state function for inheritance, a mapping from inheritance transitions to the bags of class places.

$\ddot{A}_s : T_c \rightarrow (P_c)_{MS}$ is an output state function for inheritance, a mapping from inheritance transitions to the bags of class places.

Predicate : $\{ \tilde{O}_c, \ddot{O}_c, \tilde{O}_s, \ddot{O}_s \}$ where

$\tilde{O}_c : T_r \rightarrow (P_r)_{MS}$ is an input class function for predicates, a mapping from predicates transitions to the bags of predicates.

$\ddot{O}_c : T_r \rightarrow (P_r)_{MS}$ is an output class function for predicates, a mapping from predicates transitions to the bags of predicates.

$\tilde{O}_s : T_r \rightarrow (P_r)_{MS}$ is an input state function for predicates, a mapping from predicates transitions to the bags of predicates.

$\ddot{O}_s : T_r \rightarrow (P_r)_{MS}$ is an output state function for predicates, a mapping from predicates transitions to the bags of predicates.

$C : P \rightarrow \Sigma$, a colour function, it maps each place into a colour set,

$E : A \rightarrow$ expression, an arc expression function, It is defined from A into expressions such that $\forall a \in A : [\mathrm{Type}(E(a)) = C(p(a))_{MS} \wedge \mathrm{Type}(\mathrm{Var}(E(a))) \subseteq \Sigma]$ where p(a) is the place of N(a), where $_{MS}$ stands for multi-set (or bags),

$I : P \rightarrow$ expression, an initialization function. It is defined from P into closed expressions such that: $\forall p \in P:[Type(I(p))=C(p)_{MS}]$.

4. SCCPN Representation of HES

Case I. Self-contradictory rule

Rule 1 : $A \Rightarrow C$
Rule 2 : $A' \Rightarrow \neg C$

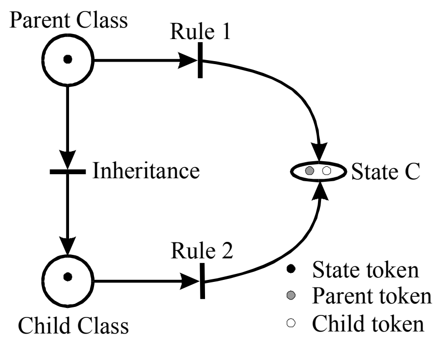

Figure 4.1

Initially, if we have a Parent token in Parent Class A is True, then Rule 1 will fire, and a Parent token will be created in State C with both A and C being True. At the same time, a Child token will be created in Child Class, having A' being True, because of inheritance. This enables Rule 2, and after firing, a Child token is also created in State C but with C' being FALSE.

Case II. Self-contradictory chain of inference

Rule 1: $B' \Rightarrow \neg C$
Rule 2 : $C' \Rightarrow D$
: :
Rule N: $N' \Rightarrow B$

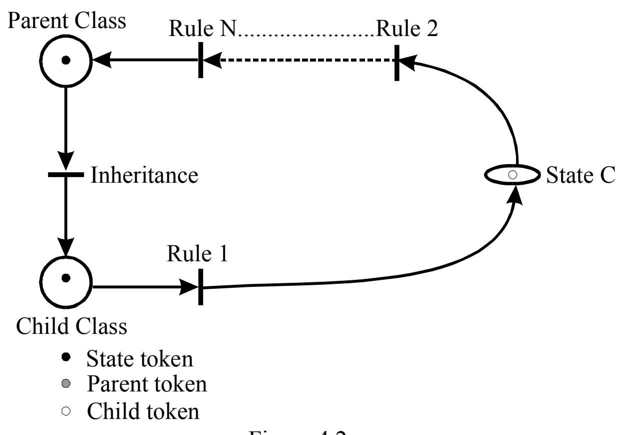

Figure 4.2.

Initially, if we have a Parent token in State C with C is True, then Rule 2 will fire, after the chain inference from Rule 2 to Rule N, a Parent token will be created in Parent Class with B being True. After inheritance, a Child token will be created in Child Class with B' being True, and this will enables Rule 1 to fire. This time, the State C is asserted to be FALSE by Rule 1 contradicting to the initial fact C which is TRUE.

Case III. Contradictory pairs of rules

Rule 1 : $A \wedge B \Rightarrow C$
Rule 2 : $A' \wedge B' \Rightarrow \neg C$

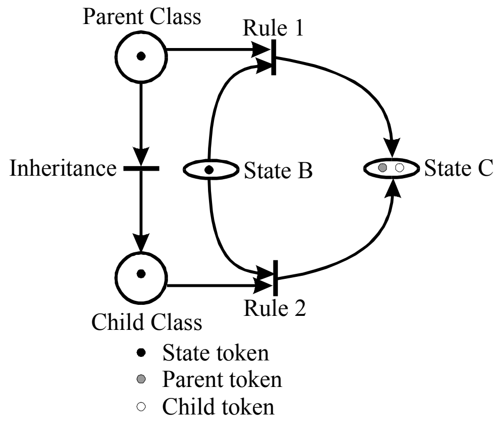

Figure 4.3.

If we have a Parent token in Parent Class with A is TRUE, and a State token in State B indicating State B is TRUE, State C will be asserted to be TRUE by Rule 1 but FALSE by Rule 2 indicating contradictory state of inference.

Case IV. Contradictory chains of rules

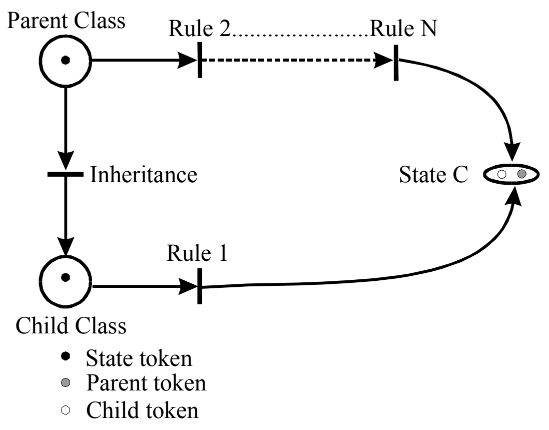

Figure 4.4.

Rule 1: $A' \Rightarrow \neg P$
Rule 2 : $A \Rightarrow B$
: :
Rule N : $N \Rightarrow P$

Proposition 4.1. For a given marking M_0, that minimally enables a nontrivial transition sequence σ_i, iff the HES has inconsistent rules causing contradiction between the parent and child object classes, then $\exists \sigma_j$, $\exists k$, such that these sequences have the following properties:

(i) $\sigma_i \cap \sigma_j = \varnothing$;
(ii) $T_c \cap \sigma_i = \varnothing$; $T_c \cap \sigma_j \neq \varnothing$;
(iii) $M' = \delta(M_0, \sigma_i)$, $M'' = \delta(M_0, \sigma_j)$;
(iv) $M_{sk}=0$, $M'_{sk}>0$, $M''_{sk}>0$;
(v) $M_{ck}=0$, $M'_{ck}>0$, $M''_{ck}>0$;
(vi) $\exists (p_{rk}, c_{ck})' \in M'_{ck}$, $\exists (p_{rk}, c_{ck})'' \in M''_{ck}$

(vii) $(p_{rk},c_{ck})'=\neg(p_{rk},c_{ck})''$

If there exists incorrect rules applied to the object hierarchy of the following cases:

Case (I): Identical Conditions but Contradict Actions between Parent Class and Child Class.

Let $E(\Phi)$ be the arc expression function of the predicate IS-A member of Parent Class and
Let $E(\phi)$ be the arc expression function of the predicate IS-A member of Child Class.

In SCCPN representation, there should exists t_{r0}, t_{r1} and t_c such that

$E(\tilde{O}_c(t_{r0}),t_{r0}) - E(\Phi)=E(\tilde{O}_c(t_{r1}),t_{r1}) - E(\phi)$
$E(t_{r0},\ddot{O}_c(t_{r0}))=\neg E(t_{r1},\ddot{O}_c(t_{r1}))$

$\tilde{O}_c(t_{r0})=\tilde{A}_c(t_c)$, $\ddot{A}_c(t_c)=\tilde{O}_c(t_{r1})$, $\ddot{O}_c(t_{r0})=\ddot{O}_c(t_{r1})$,
$\tilde{O}_s(t_{r0})=\tilde{A}_s(t_c)$, $\ddot{A}_s(t_c)=\tilde{O}_s(t_{r1})$, $\ddot{O}_s(t_{r0})=\ddot{O}_s(t_{r1})$

Choose M_0 with a class token element (p_{r0},c_{c0}) and a state token (p_{r0},c_{s0}) s.t. t_{r0} is minimally enabled,

then $M_{sk}=1$ if $p_{sk}\in\tilde{O}_s(t_{r0})$, 0 otherwise.
And $M_{ck}=1$ if $p_{ck}\in\tilde{O}_c(t_{r0})$, 0 otherwise.

Since $\tilde{O}_s(t_{r0})=\tilde{A}_s(t_c)$ and $\tilde{O}_c(t_{r0})=\tilde{A}_c(t_c)$, t_c is enabled.
Since t_c is enabled, the new marking in $\ddot{A}_c(t_c)=1$ and has a colour of (p_{r1},c_{c1}) which is inherited from (p_{r0},c_{c0}). Where $E(p_{r0},t_c) - E(\Phi)=E(t_c,p_{r1}) - E(\phi)$.
Since $E(\tilde{O}_c(t_{r1}),t_{r1})=E(\tilde{O}_c(t_{r0}),t_{r0}) - E(\Phi) + E(\phi)$, therefore t_{r1} is enabled.

Therefore, $\exists M'$, $\exists M''$ s.t. $M'=\delta(M_0,\sigma_i)$, $M''=\delta(M_0,\sigma_j)$ and $\sigma_i=(t_{r0})$, $\sigma_j=(t_c,t_{r1})$.

Therefore

$$M'_{sk}=\begin{cases}1 & \text{if } p_{sk}\in\{\tilde{O}_s(t_{r0}),\ddot{O}_s(t_{r0})\}\\ 0 & \text{otherwise}\end{cases}$$

$$M'_{ck}=\begin{cases}1 & \text{if } p_{ck}\in\ddot{O}_c(t_{r0})\\ 0 & \text{otherwise}\end{cases}$$

And the colour of the class token at $\ddot{O}_c(t_{r0})=(p_{rk},c_{ck})'$

$$M''_{sk}=\begin{cases}1 & \text{if } p_{sk}\in\{\tilde{O}_s(t_{r1}),\ddot{O}_s(t_{r1})\}\\ 0 & \text{otherwise}\end{cases}$$

$$M''_{ck}=\begin{cases}1 & \text{if } p_{ck}\in\ddot{O}_c(t_{r1})\\ 0 & \text{otherwise}\end{cases}$$

And the colour of the class token at $\ddot{O}_c(t_{r1})=(p_{rk},c_{ck})''$

Since $E(t_{r0},\ddot{O}_c(t_{r0}))=\neg E(t_{r1},\ddot{O}_c(t_{r1}))$,
therefore $(p_{rk},c_{ck})'=\neg(p_{rk},c_{ck})''$

Thus, for $p_{sk}\in\ddot{O}_s(t_{r0})$, $M_{sk}=0$, $M'_{sk}>0$, $M''_{sk}>0$, and for $p_{ck}\in\ddot{O}_c(t_{r0})$, $M_{ck}=0$, $M'_{ck}>0$, $M''_{ck}>0$, and $(p_{rk},c_{ck})'=\neg(p_{rk},c_{ck})''$, implying inconsistency with $\sigma_i=(t_{r0})$, $\sigma_j=(t_c,t_{r1})$ in the object classes.

Case (II): Contradictory pair of rules between Parent Class and Child Class.

Let $E(\Phi)$ be the arc expression function of the predicate IS-A member of Parent Class and
Let $E(\phi)$ be the arc expression function of the predicate IS-A member of Child Class.

In SCCPN representation, there should exists t_{r0}, t_{r1} and t_c such that

$\Sigma\{E(\tilde{O}_c(t_{r0}),t_{r0})\} - E(\Phi)=\Sigma\{E(\tilde{O}_c(t_{r1}),t_{r1})\} - E(\phi)$
$E(t_{r0},\ddot{O}_c(t_{r0}))=\neg E(t_{r1},\ddot{O}_c(t_{r1}))$

$\tilde{A}_c(t_c)\subset\tilde{O}_c(t_{r0})$, $\ddot{A}_c(t_c)\subset\tilde{O}_c(t_{r1})$, $\ddot{O}_c(t_{r0})=\ddot{O}_c(t_{r1})$,
$\tilde{A}_s(t_c)\subset\tilde{O}_s(t_{r0})$, $\ddot{A}_s(t_c)\subset\tilde{O}_s(t_{r1})$, $\ddot{O}_s(t_{r0})=\ddot{O}_s(t_{r1})$

Choose M_0 with a class token element (p_{r0},c_{c0}) and a state token (p_{r0},c_{s0}) s.t. t_{r0} is minimally enabled,

then $M_{sk}=1$ if $p_{sk}\in\tilde{O}_s(t_{r0})$, 0 otherwise.
And $M_{ck}=1$ if $p_{ck}\in\{\tilde{O}_c(t_{r0})\cap\tilde{A}_c(t_c)\}$, 0 otherwise.

Since $\tilde{A}_s(t_c)\subset\tilde{O}_s(t_{r0})$ and $\tilde{A}_c(t_c)\subset\tilde{O}_c(t_{r0})$, t_c is enabled.
Since t_c is enabled, the new marking in $\ddot{A}_c(t_c)=1$ and has a colour of (p_{r1},c_{c1}) which is inherited from (p_{r0},c_{c0}). Where $E(p_{r0},t_c) - E(\Phi)=E(t_c,p_{r1}) - E(\phi)$.
Since $\Sigma\{E(\tilde{O}_c(t_{r0}),t_{r0})\} - E(\Phi)=\Sigma\{E(\tilde{O}_c(t_{r1}),t_{r1})\} - E(\phi)$, therefore t_{r1} is enabled.

Therefore, $\exists M'$, $\exists M''$ s.t. $M'=\delta(M_0,\sigma_i)$, $M''=\delta(M_0,\sigma_j)$ and $\sigma_i=(t_{r0})$, $\sigma_j=(t_c,t_{r1})$.

Therefore

$$M'_{sk}=\begin{cases}1 & \text{if } p_{sk}\in\{\tilde{O}_s(t_{r0}),\ddot{O}_s(t_{r0})\}\\ 0 & \text{otherwise}\end{cases}$$

$$M'_{ck}=\begin{cases}1 & \text{if } p_{ck}\in\ddot{O}_c(t_{r0})\\ 0 & \text{otherwise}\end{cases}$$

And the colour of the class token at $\ddot{O}_c(t_{r0})=(p_{rk},c_{ck})'$

$$M''_{sk}=\begin{cases}1 & \text{if } p_{sk}\in\{\tilde{O}_s(t_{r1}),\ddot{O}_s(t_{r1})\}\\ 0 & \text{otherwise}\end{cases}$$

$$M''_{ck}=\begin{cases}1 & \text{if } p_{ck}\in\ddot{O}_c(t_{r1})\\ 0 & \text{otherwise}\end{cases}$$

And the colour of the class token at $\ddot{O}_c(t_{r1})=(p_{rk},c_{ck})''$

Since $E(t_{r0},\ddot{O}_c(t_{r0}))=\neg E(t_{r1},\ddot{O}_c(t_{r1}))$,
therefore $(p_{rk},c_{ck})'=\neg(p_{rk},c_{ck})''$

Thus, for $p_{sk}\in\ddot{O}_s(t_{r0})$, $M_{sk}=0$, $M'_{sk}>0$, $M''_{sk}>0$, and for $p_{ck}\in\ddot{O}_c(t_{r0})$, $M_{ck}=0$, $M'_{ck}>0$, $M''_{ck}>0$, and

$(p_{rk},c_{ck})'=\neg(p_{rk},c_{ck})''$, implying inconsistency with $\sigma_i=(t_{r0})$, $\sigma_j=(t_c,t_{r1})$.

Case(III): Contradictory chains of rules between the parent and child object classes.

Let $E(\Phi)$ be the arc expression function of the predicate IS-A member of Parent Class and
Let $E(\phi)$ be the arc expression function of the predicate IS-A member of Child Class.

In SCCPN representation, there should exists $\sigma_i=(t_{r1},t_{r2},\ldots t_{rj})$ and $\sigma_j=(t_c,t_{r0})$ such that

$E(\tilde{O}_c(t_{r0}),t_{r0}) - E(\Phi)=E(\tilde{O}_c(t_{r1}),t_{r1}) - E(\phi)$
$E(t_{r0},\ddot{O}_c(t_{r0}))=\neg E(t_{rj},\ddot{O}_c(t_{rj}))$

$\ddot{O}_s(t_{r(m)})=\tilde{O}_s(t_{r(m+1)})$ for m=1,2,…..j-1.
$\ddot{O}_c(t_{r(m)})=\tilde{O}_c(t_{r(m+1)})$ for m=1,2,…..j-1.
$\tilde{O}_c(t_{r1})=\tilde{A}_c(t_c)$, $\ddot{A}_c(t_c)=\tilde{O}_c(t_{r0})$,
$\tilde{O}_s(t_{r1})=\tilde{A}_s(t_c)$, $\ddot{A}_s(t_c)=\tilde{O}_s(t_{r0})$,

Choose M_0 with a class token element (p_{r0},c_{c0}) and a state token (p_{r0},c_{s0}) s.t. $\sigma_i=(t_{r1},t_{r2},\ldots t_{rj})$ is minimally enabled, i.e., $\forall$m=1,2,3,….j-1,

$$\text{then } M_{sk}=\begin{cases}1 & \mathit{if}\ p_{sk}\in\tilde{A}_s(t_c)\\ 0 & \mathit{otherwise}\end{cases}$$

$$\text{And } M_{ck}=\begin{cases}1 & \mathit{if}\ p_{ck}\in\tilde{A}_c(t_c)\\ 0 & \mathit{otherwise}\end{cases}$$

The execution of transition sequence, σ_i, gives M' s.t. $\forall$m=1,2,3,….j, $\ddot{O}_s(t_m)\in\ddot{O}_s(\sigma_i)$

$$M'_{sk}=\begin{cases}1 & \mathit{if}\ p_{sk}\in\{\tilde{O}_s(t_{r1}),\ddot{O}_s(\sigma_i)\}\\ 0 & \mathit{otherwise}\end{cases}$$

And the colour of the class token at $\ddot{O}_c(t_j)=(p_{rk},c_{ck})'$

Since $\tilde{A}_s(t_c)=\tilde{O}_s(t_{r1})$ and $\tilde{A}_c(t_c)=\tilde{O}_c(t_{r1})$, t_c is enabled.
Since t_c is enabled, the new marking in $\ddot{A}_c(t_c)=1$ and has a colour of (p_{r1},c_{c1}) which is inherited from (p_{r0},c_{c0}). Where $E(p_{r1},t_c) - E(\Phi)=E(t_c,p_{r0}) - E(\phi)$.
Since $E(\tilde{O}_c(t_{r1}),t_{r1}) - E(\Phi)=E(\tilde{O}_c(t_{r0}),t_{r0}) - E(\phi)$, therefore t_{r0} is enabled.

Let $M''_{ck}=\delta(M'_{ck},t_{r0})$,

$$M''_{ck}=\begin{cases}1 & \mathit{if}\ p_{ck}\in\ddot{O}_c(t_{r0})\\ 0 & \mathit{otherwise}\end{cases}$$

And the colour of the class token at $\ddot{O}_c(t_{r0})=(p_{rk},c_{ck})''$

$E(t_{r0},\ddot{O}_c(t_{r0}))=\neg E(t_{rj},\ddot{O}_c(t_{rj}))$, therefore $(p_{rk},c_{ck})'=\neg(p_{rk},c_{ck})''$

Case (IV): Self Contradictory chain of inference between the parent and child object classes.

Proposition 4.2. For a given marking M^0, that minimally enables transition sequence α, iff the HES has inconsistent rules causing self-contradictory chain of inference between the parent and child object classes, then $\exists$j, $\exists$k such that the sequence has the following properties:

(i) $M^i\in[M^0>=\{M^0,M^1,M^2,\ldots M^i,..M^j\}$,
(ii) $M^j=\delta(M^0,\alpha)$ for j>0,
(iii) $T_c\cap\alpha\neq\varnothing$;
(iv) $M_{sk}[\tilde{O}]=0$, $M^0_{sk}[\tilde{O}]>0$, $M^j_{sk}[\tilde{O}]>1$.
(v) $\exists(p_{rk},c_{ck})'\in M^0_{ck}$, $\exists(p_{rk},c_{ck})''\in M^j_{ck}$
(vi) $(p_{rk},c_{ck})'=\neg(p_{rk},c_{ck})''$

In SCCPN representation, there should exist $\alpha=(t_{r1},t_{r2},\ldots t_{r(l-1)},t_c,t_{rl},\ldots t_{rm})$ forming a connected path such that

$\tilde{O}_s(t_{r(l+1)})\subseteq\ddot{O}_s(t_{rl})$ for l=1,2,.....m-1,
$\tilde{O}_s(t_{r1})\subseteq\ddot{O}_s(t_{rm})$.
$E(\tilde{O}_c(t_{r1}),t_{r1})=\neg E(t_{rm},\ddot{O}_c(t_{rm}))$

Choose M^0 with a class token element $(p_{r0},c_{c0})'$ and a state token $(p_{r0},c_{s0})'$ s.t. $\alpha=(t_{r1},t_{r2},\ldots t_{r(l-1)},t_c,t_{rl},\ldots t_m)$ is minimally enabled,i.e., $\forall l$=1,2,....m-1,

then $M_{sk}=1$ if $p_{sk}\in\tilde{O}_s(t_{r1})$, 0 otherwise.
And $M_{ck}=1$ if $p_{ck}\in\tilde{O}_c(t_{r1})$, 0 otherwise.

$$M^0_{sk}=\begin{cases}1 & \mathit{if}\ p_{sk}\in\tilde{O}_s(t_{r1}),\mathit{where}\ M^0_{ck}(p_{ck})=1\\ 0 & \mathit{otherwise}\end{cases}$$

i.e. $M^0_{sk}[\tilde{O}_s(t_{r1})]=1$ if $p_{sk}\in\tilde{O}_s(t_{r1})$
Since $\tilde{O}_s(t_{r1})\subseteq\ddot{O}_s(t_{rm})$, and $M^m=\delta(M^0,\alpha_i)$. Therefore the execution of transition sequence, α, gives M^m s.t. $\forall l$=1,2,....m-1.

$$M^m_{sk}=\begin{cases}2 & \mathit{if}\ p_{sk}\in\tilde{O}_s(t_{r1})\\ 1 & \mathit{if}\ p_{sk}\in\{\ddot{O}_s(t_{r1}),\ \ddot{O}_s(t_{rm})\}\\ 0 & \mathit{otherwise}\end{cases}$$

And the colour of the class token at $\ddot{O}_c(t_{rm})=(p_{rk},c_{ck})''$

Since $E(\tilde{O}_c(t_{r1}),t_{r1})=\neg E(t_{rm},\ddot{O}_c(t_{rm}))$
therefore $(p_{rk},c_{ck})'=\neg(p_{rk},c_{ck})''$

Thus for $p_{sk}\in\tilde{O}_s(t_{r1})$, $M_{sk}[\tilde{O}]=0$, $M^0_{sk}[\tilde{O}]>0$, $M^j_{sk}[\tilde{O}]>1$, $\exists(p_{rk},c_{ck})'\in M^0_{ck}$, $\exists(p_{rk},c_{ck})''\in M^j_{ck}$ and $(p_{rk},c_{ck})'=\neg(p_{rk},c_{ck})''$, implying inconsistent rules causing self-contradictory chain of inference between the parent and child object classes.

5. SUMMARY AND CONCLUSION

In this paper, we have described a formal description technique based on State Controlled Coloured Petri Nets to model hybrid (rule- and frame-based) expert systems. The technique allows the use of reachability theory for the verification of the systems. The paper illustrates the capability of the technique to identify the anomalies due to the contradictions of the hybrid knowledge base. The verification was done exhaustively by minimally initiating any sequence of transitions and closely examining the reachability markings at each transition. Propositions are formulated to verify errors and anomalies in HES.

Future work will include measuring and analyzing the state-space complexity of HES and evaluating our approach for modelling and verification. We would also like to investigate further the capability of the methodology to handle fuzzy and temporal expert systems.

6. REFERENCES

[1] Agarwal, R. & Tanniru, M. (1992). A Petri Net based approach for verifying the integrity of production systems. *International Journal of Man Machine Studies*. Vol. 36. 447-468.

[2] Aikins, J.S. (1993). Prototypical Knowledge for Expert Systems: a retrospective analysis. In Bobrow D.G. (Ed.) *Artificial Intelligence*. Vol. 59. pp. 207-211. Elsevier, Amsterdam.

[3] Coenen, F. & Bench-Capon, T. (1993). *Maintenance of Knowledge-based Systems.* Academic Press.

[4] Dori, D. & Tatcher, E. (1994). Selective multiple inheritance. *IEEE Software*. Vol. 11. No. 3, 77-85.

[5] Durkin, J. (1994). *Expert Systems: Design and Development*. Macmillan Publishing Company. 12-23;711-771.

[6] French, S.W. & Hamilton, D. (1994). A Comprehensive Framework for Knowledge-Base Verification and Validation. International *Journal of Intelligent Systems*. Vol. 9. 809-837.

[7] Gamble, R. F. & Baughman D. M. (1996). A methodology to incorporate formal methods in hybrid KBS verification. *International Journal of Human Computer Studies*. Vol. 44, 213-244.

[8] Gamble, R.F., Roman G., Ball W.E. & Cunningham H.C. (1994). Applying Formal Verification Methods to Rule-Based Programs. *International Journal of Expert Systems*. Vol. 7, no. 3, 203-239.

[9] Gupta, U. (Ed.) (1991). *Validating and Verifying Knowledge-based Systems*. IEEE Computer Society Press.

[10] Jensen, K. (1995). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*. Vol 2. Springer-Verlag.

[11] Jensen, K. (1996). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use.* Vol. 1. 2nd Ed. Springer-Verlag.

[12] Lee, S. & O'Keefe, R.M. (1993). Subsumption Anomalies in Hybrid Knowledge Bases. *International Journal of Expert Systems*. Vol. 6, No. 3, 299-320.

[13] Liu, N. K. & Dillon, T. (1995). Formal Description and Verification of Production Systems. *International Journal of Intelligent Systems*. Vol. 10, 399-442.

[14] Murrell, S. & Plant, R. (1996). On the Validation and Verification of Production Systems: a graph reduction approach. *International Journal of Human Computer Studies*. Vol. 44, 127-144.

[15] Nazareth, D. L. (1993). Investigating the Applicability of Petri Nets for Rule-Based System Verification. *IEEE Transactions on Knowledge and Data Engineering*. Vol. 4, No. 3, 402-415.

[16] O'Keefe, R.E. & O'Leary, D.E. (1993). Expert System Verification and Validation: A survey and tutorial. *Artificial Intelligence Review*. Vol. 7, 3-42.

[17] Scarpelli, H. & Gomide, F. (1994). A high level net approach for discovering potential inconsistencies in fuzzy knowledge bases. *Fuzzy Sets and Systems*. Vol. 64, 175-193.

[18] Shiu, S., Liu, J. & Yeung, D. (1995a). Modelling Hybrid Rule/Frame-based Expert Systems Using Coloured Petri Nets. In *Proceedings of 8th International Conference on Industrial & Engineering Applications of AI & ES*. Melbourne, Australia. 525-532.

[19] Shiu, S., Liu, J. & Yeung, D. (1995b). An Approach Towards the Verification of Hybrid Rule/Frame-based Expert Systems using Coloured Petri Nets. In *Proceedings of 1995 IEEE International Conference on SMC*. Vancouver. 2257-2262.

[20] Shiu, S., Liu, J. & Yeung, D. (1996a). An Approach Towards the Verification of Fuzzy Hybrid Rule/Frame-based Expert Systems using Coloured Petri Nets. In *Proceedings of ECAI-96 Workshop in Validation, Verification and Refinement of KBS*. Budapest, 105-113.

[21] Shiu, S., Liu, J. & Yeung, D. (1996b). Proofs of the Formal Verification of Hybrid Rule/Frame-based Expert Systems using SCCPN. *Unpublished Manuscript*. Department of Computing, Hong Kong Polytechnic University.

[22] Vranes, S. & Stanojevic, M. (1995). Integrating Multiple Paradigms within the Blackboard Framework. *IEEE Transactions on Software Engineering*. Vol. 21, No. 3, 244-262.

[23] Willis, C.P. (1996). Analysis of inheritance and multiple inheritance. *Software Engineering Journal*. July.

[24] Zhang, D. & Nguyen D. (1994). PREPARE: A Tool for Knowledge Base Verification. *IEEE Transactions on Knowledge and Data Engineering*. Vol. 6, No. 6, December, 983-989.