派翠網路模擬分析系統的物件導向發展

Object-Oriented Development of a Petri Nets Modeling and Analysis System

楊鎮華
Stephen J.H. Yang
Department of Computer Science and Information Engineering, National Central University, Taiwan.
jhyang@se01.csie.ncu.edu.tw

劉豐瑞
Kevin F.R. Liu
Department of Civil Engineering, National Central University, Taiwan.
frliu@se01.csie.ncu.edu.tw

林聖博
Sam Lin
Department of Computer Science and Information Engineering, National Central University, Taiwan.
sam@se01.csie.ncu.edu.tw

李允中
Jonathan Lee
Department of Computer Science and Information Engineering, National Central University, Taiwan.
yjlee@se01.csie.ncu.edu.tw

黃為德
Wei T. Huang
Department of Computer Science and Information Engineering, National Central University, Taiwan.
wthuang@csie.ncu.edu.tw

摘 要

本論文以物件導向程式語言-Java Applet實作派翠網路系統(Petri Nets Modeling and Analysis System, PNMAS)。PNMAS實作一個派翠網路與兩個模糊派翠網路，具有繪圖、設定、執行與分析的功能。

Abstract

In this paper, we are presenting the implementation of Petri Nets Modeling and Analysis System (PNMAS) using Java Applet. The users of PNMAS are capable of drawing , setting, executing and analyzing classical Petri nets and two fuzzy Petri nets. PNMAS is now available on the Internet via http://140.115.50.137.

Keywords: Petri nets, fuzzy Petri nets, object-oriented programming, Java Applet.

1 Introduction

Petri nets are a graphical and mathematical tool for modeling and studying systems. The graphical nature of Petri nets is used as a visual-communication to simulate the dynamic and concurrent activities of systems, and the mathematical power make it possible to govern the behavior of systems. After modeled by a Petri net, a system can be revealed more deeper insight about its structure and dynamic behavior by analyzing the Petri net.

Recently, Petri nets are utilized for modeling fuzzy rule-based reasoning in order to improve the efficiency of rule-based reasoning [buga94, chen90, kona96, loon88, scar96a, scar96b, yang97, liu97]. The main advantages of using Petri nets for modeling fuzzy rule-based reasoning are summarized as follows: First, Petri net's graphical representation can help experts construct and modify fuzzy rule bases. Second, the graphical nature of Petri nets is suitable for visualizing sequences of transition firing that can be utilized as an explanation tool for rule-based systems. Third, the mathematical foundation of Petri nets can express structural and dynamic behaviors in algebraic forms, i.e., a system of linear equations. Once rules have been transformed into Petri nets, the fuzzy reasoning problems can be transformed into the linear equation problems which can be solved in parallel. Fourth, Petri nets have well established formal mechanisms for modeling and property checking of concurrent and synchronization structures and such formalism can be used for achieving the and/or parallelism of rule-based systems. Fifth, Petri net's analytic capability can help with checking properties of modeled systems for gaining deeper insights into the systems.

In this paper, we are presenting the implementation of Petri Nets Modeling and Analysis System (PNMAS) using Java Applet. The users of PNMAS are capable of drawing , setting, executing and analyzing classical Petri nets and two fuzzy Petri nets. We also propose the notion of hierarchy to manage the complexity of Petri nets. PNMAS consists of four subsystems: files managing subsystem, drawing subsystem, simulation subsystems and analysis subsystem. Files managing subsystem deals with saving/loading files. Drawing subsystem is in charge of drawing and setting Petri nets. Simulation subsystem is to compute the fuzzy values and move tokens after judging firing conditions. Analysis subsystem analyzes

the properties of Petri nets.

The organization of this paper is as follows. The definition of Petri nets and fuzzy Petri nets are briefly described in next section. In section 3, the implementation of PNMAS is described, such as the organization of PNMAS and the proceduses of implementation. An example which illustrates how to use PNMAS to execute the fuzzy Petri nets for modeling fuzzy rule-based reasoning is shown in section 4. Finally, a summary of this paper is given in the section 5.

2 Petri Nets and Fuzzy Extensions of Petri Nets

The Petri Nets Modeling and Analysis System (PNMAS) is currently developed based on three Petri net-based theories - the classical Petri nets [mura89] and two kinds of fuzzy extensions of Petri nets [yang97, liu97]. In this section, we introduce the three theories as follows.

2.1 Petri Nets

A Petri net is a directed, weighted, bipartite graph consisting of two kinds of nodes, called places (p_i) and transitions (t_j), where arcs are either from a place to a transition or from a transition to a place. Murata [mura89] has formally defined Petri nets as a 5-tuple: PN=(P,T,F,W,M_0), where $P=\{p_1, p_2,...,p_m\}$ is a finite set of places, $T=\{t_1, t_2,..., t_n\}$ is a finite set of transitions, $F\subseteq(P\times T)\cup(T\times P)$ is a set of arcs, W: $F\rightarrow\{1,2,3,...\}$ is a weight function, and M_0: $P\rightarrow\{1,2,3,...\}$ is the initial marking. A marking M is a m-vector, $\{M(p_1),..., M(p_m)\}$, where $M(p_i)$ denotes the number of the tokens in place p_i. The incidence matrix $A=[a_{ij}]$ is a n×m matrix of integers and its typical entry is defined by $a_{ij}= a_{ij}^+ - a_{ij}^-$, where a_{ij}^+ is the weight of the arc from a transition t_i to its output place p_j, and a_{ij}^- is the weight of the arc to a transition t_i from its input place p_j. The evolution of markings, used to simulate the dynamic behavior of a system, is based on the firing rule, such as: a transition t is enabled if each input place t is marked with at least $w(p,t)$ tokens, where $w(p,t)$ is the weight of the arc from p to t; an enabled transition may or may not enabled; and a firing of an enabled transition t removes $w(p,t)$ tokens from each input place p of t, and adds $w(t,p)$ tokens to each output place p of t, where $w(t,p)$ is the weight of the arc from t to p.

2.2 Fuzzy Extensions of Petri Nets

A typical interpretation of Petri nets is to view a place as a condition, a transition as the causal relationship of conditions, and a token in a place as a fact used to claim the truth of the condition associated with the place. The real-world problems often contain fuzzy or uncertain information, for example: (1) conditions are fuzzy; (2) the causal relationship of fuzzy conditions are uncertain; (3) the values of facts are fuzzy, and may partially match the value of the related condition; and (4) the confidence about the truths of the facts is not complete.

In [yang97], the authors defined fuzzy Petri nets 9-tuple: FPN=(P,T,A,M,C,O,F,B,L), where P is a set of places, i.e., $P=\{p_1, p_2,...,p_m\}$, T is a set of transitions, i.e., $T=\{t_1, t_2,..., t_n\}$, A is a n×m matrix in which a_{ij}^+ is the weight of the arc from a transition t_i to its output place p_j, and a_{ij}^- is the weight of the arc to a transition t_i from its input place p_j, W is a set of marking with m-vector, $\{M(p_1),..., M(p_m)\}$, where $M(p_i)$ denotes the numbers of the tokens in place p_i, M_0 denotes the initial marking, C is a set of mapping between rules and transitions, i.e., $C(r_i)=t_i$, $C(t_i)=r_i$, O is a set of mapping between antecedent/consequent parts and places, i.e., $O(s_i)=p_i$, $C(p_i)=s_i$, F is a set of mapping from the places to their corresponding degree of truth, i.e., $F(p_i)=\alpha_i$, B is a set of mapping from the transitions to their corresponding degree of truth, i.e., $B(t_i)=\beta_i$, L is set of thresholds of transition firing; $L(t_i)=\lambda_i$. Let $^\bullet t_j$ denotes the input places of t_j, and $t_j^\bullet$ denotes the output places of t_j. For a transition t_j in FPN to be enable, the degree of truth of all t_j's input places must be greater than or equal to t_j's threshold. That is, a transition is enabled when $a_i \geq \lambda_j$, where $a_i \in {}^\bullet t_j$ and in $\lambda_j \in [0,1]$. Thus, FPN is a generalized version of PN. When all transitions t_j in a FPN with $\lambda_j = 1$, a FPN becomes a PN.

In [liu97], the authors defined fuzzy Petri nets as a 5-tuple: FPN=(FP,UT,F,W,M_0). $FP=\{(p_1, F_1), (p_2, F_2),..., (p_m, F_m)\}$ is a finite set of fuzzy places, where p_i represent a fuzzy condition and F_i is a fuzzy subset of U_i to represent the fuzzy set of the condition. $UT=\{(t_1, \tau_1), (t_2, \tau_2),..., (t_n, \tau_n)\}$ is a finite set of uncertain transitions, where t_j represents the causal relationship of fuzzy conditions and τ_j is a fuzzy truth value to represent the uncertainty about the causal relationship of fuzzy conditions. $F\subseteq(FP\times UT)\cup(UT\times FP)$ is a set of arcs. W: $F\rightarrow\{1,2,3,...\}$ is a weight function, and M_0: $P\rightarrow\{1,2,3,...\}$ is the initial marking. The fuzzy truth value serves as the representation of uncertainty for its capability to express the possibility of the degree of truth. Each token is associated with a pair of fuzzy sets (F_i', τ_i). A firing of an enabled uncertain transition t_j removes the uncertain fuzzy token from each input place p_i of t_j, adds a new token to each output place p_k of t_j, and the fuzzy set and fuzzy truth value attached to the new token will be computed based on the mechanism in fuzzy reasoning.

3 Implementation

The Petri Nets Modeling and Analysis System is implemented by Java Applet, an object-oriented programming language by Sun. Java is chosen as our programming language because it is a cross platform and secure language.

3.1 Architecture

The architecture of network for PNMAS is shown in

Figure 1. We design a Petri net server together with Alibaba (a www server) to help with transmitting files under Client/Server mode. In Figure 1, client 1 is a workstation with UNIX and client 2 is a PC with Window 95. To avoid problems arising from saving or loading files among clients, only one client who takes token can process files at the same time..

3.2 PNMAS Organization

PNMAS consists of four subsystems: files managing subsystem, drawing subsystem, simulation subsystems and analysis subsystem. The subsystems are described as follows. First, files managing subsystem, consisting of Petri Nets server and Alibaba, deals with saving/loading files. Second, drawing subsystem is in charge of drawing and setting Petri nets, such as drawing places, transitions , arcs and setting fuzzy values. Third, simulation subsystem is to compute the fuzzy values and move tokens after judging firing conditions. Fourth, analysis subsystem analyzes the properties of drawing Petri nets, such as incidence matrix, reachability graph and state equation.

3.3 Hierarchy

We are confronted with the problem of complexity when the Petri nets are too large to manage. Even though a simple system is modeled by Petri net, the Petri net may be quite large. Therefore, we propose the notion of hierarchy to manage the complexity of Petri nets. Besides, hierarchy is used to structuralize Petri nets, similar to the design of module in the rule bases of expert systems. Hierarchy, represented by a double-lined square, means acyclic Petri net. Hierarchy is linked to places which can be viewed as ports. As shown in Figure 2, H_1 and H_2 are two hierarchies, where accommodate Petri nets, place p_1 is considered as the output port of hierarchy H_1, and place p_2 is considered as the input port of hierarchy H_2. Once place p_b in hierarchy H_1 receives tokens, the tokens will be sent to place p_1. Similarly, Once place p_2 receives tokens, the tokens will be sent to place p_a in hierarchy H_2.

There are three main benefits by having a hierarchical structure in our system: (1) the notion of hierarchy can make easy the handling of complex systems through decomposition; (2) a hierarchical Petri net can facilitate the reusability; and (3) a hierarchy is an application of the notion of system architecture [luck95]. We can consider a hierarchy as an interface. A Petri net in a hierarchy is considered as a module. Arcs in a Petri net having hierarchies is considered as the connections between interfaces. Transitions is considered as formal constraint. Therefore, a Petri net having hierarchies is considered as a system architecture. A system architecture without module is called instantiated architecture (see Figure 3).

3.4 Procedures of Implementation:

1. Defining classes: Objects are classified into three categories [ivar92]: interface objects, entity objects and control objects. They are described as follows: (1) Interface objects are everything concerning any interface to PNMAS. An example of an interface object is the user interface functionality for requesting information about Petri nets. These objects are to transform users' input into events or to show information requested by users. (2) Entity objects model information in PNMAS that should be held for a long time. Examples of entity objects are places, transitions, arcs, hierarchies and tokens. (3) Control objects model functionality that is not naturally tied to any other object, for example, doing some analysis of Petri nets and then returning the result to an interface object.
2. Defining operation functions: Operation functions are defined for each classes based on the classes state and behavior.
3. Implementing operation functions: Operation functions are implemented based on their definitions. An operation function will be called and carried out if specific event occurs.
4. Defining subsystems: The classesects which serve the same functionality are collected into a subsystem. There are four subsystems in PNMAS: Files managing subsystem, drawing subsystem, simulation subsystems and analysis subsystem.
5. Implementing subsystems:
   (a) Files managing subsystem: Files managing subsystem, consisting of Petri Nets server and Alibaba, is in charge of processing files, such as saving and loading files. Files managing subsystem allows at most ten users to login PNMAS, but only one user who takes token can save or load files at the same time. Besides, the mechanism to deal with errors is defined as follows: (1) The subsystem will end the login of a user if the user disconnect PNMAS. (2) An error message will be sent to users if there is something wrong with the servers.
   (b) Drawing subsystem: First, the two fuzzy Petri nets inherit from the classical Petri because they both stick to the basic characteristics of classical Petri nets (see Figure 4a). Second, the inheritance between five basic classes (i.e. places, transitions, arcs, hierarchies and tokens) should be specified (see Figure 4b). In Figure 4b, places, transitions and hierarchies are collected into a PN_Classes because they have some similar characteristics, where PN_Classes take charge of scaling the objects. Third, the classes for customer windows are divided into two groups: tool classes and canvas classes (see Figure 5). In Figure 5, action() function is used

to transmit information from a lower level to an upper level. For example, if button 1 is pushed then a relevant event will be sent to the action() of tool classes. After processed in tool classes, the information is sent to customer window classes and then the drawing will be executed. It is helpful to debug, maintain and modify PNMAS that way instead of directly calling the functions of classes.

(c) Simulation subsystem: There are two simulation modes. One is the firings of transitions and the other is the firings of hierarchies. First, the firings of transitions are based the enabling and firing rules described in section 2. Second, to increase the performance of PNMAS, the firings of hierarchies are based on the following conditions: (1) The Petri net in a hierarchy is enabled in the initial marking. (2) Tokens are put into a hierarchy by users and make it enabled. (3) The input ports of a hierarchy receive tokens.

(d) Analysis subsystem: There are three analysis classes: Incidence matrix classes, reachability graph classes and state equation classes.

The PNMAS is now available on the Internet via http://140.115.50.137.

4 An Example

This section illustrates how to use PNMAS to execute the fuzzy Petri nets proposed by Liu [liu97] for modeling fuzzy rule-based reasoning. Assume that there is a rule base containing 5 truth-qualified fuzzy rules, one module having 2 truth-qualified fuzzy rules, and three truth-qualified fuzzy facts as follows:

Main Rules:

Rule 1: IF (X_1 is large) AND (X_2 is small) THEN (X_3 is large), very true

Rule 2: IF (X_2 is very small) THEN (X_4 is very small), true

Rule 3: IF (X_5 is large) THEN (X_6 is very large), true

Rule 4: IF (X_7 is large) THEN (X_8 is large), very true

Rule 5: IF (X_9 is small) THEN (X_8 is very large), very true

Module 1:

Rule 1: IF (X_3 is large) AND (X_4 is small) THEN (X_7 is very large), true

Rule 2: IF (X_6 is large) THEN (X_9 is very small), very true

Facts:

Fact 1: (X_1 is fairly large), very true

Fact 2: (X_2 is very small), true

Fact 3: (X_5 is very large), very true

where X_i is a variable; very large, large, fairly large, very small, small, fairly small are fuzzy sets of the universe of discourse from 0 to 1 to represent fuzziness; very true, true, fairly true, very false, false, fairly false are fuzzy truth values to represent the uncertainty.

The rules, module and facts are input to PNMAS and then they are automatically transformed into a fuzzy Petri net containing a hierarchy through our proposed algorithms (see Figure 6). The hierarchy represents a fuzzy Petri net mapping onto on the module in the rule base (see Figure 7). In Figure 6, the fuzzy proposition "X_2 is small" in Rule 2 is transformed into a fuzzy place (p_4, S)(i.e. S: small), the fuzzy proposition "X_4 is very small" in Rule 2 is transformed into a fuzzy place (p_5, VS)(i.e. VS: very small), and Rule 2 is represented by the inference transition (t_2, T)(i.e. T: true). Fact 1 is transformed into a token deposited in p_1, and its value is (FL, VT)(i.e. FL: fairly large) which is shown after clicking p_1 twice. After pushing the run button, the dynamic behavior of the fuzzy rule-based reasoning is animatedly shown in the window by moving tokens. The result is shown in Figure 8. Figure 9 is clicked from p_{13} in Figure 8 to show the detail information about the result.

5 Conclusion

We have implemented Petri Nets Modeling and Analysis System (PNMAS) using Java Applet. We have also propose the notion of hierarchy to reduce the complexity of Petri nets and to structuralize Petri nets. PNMAS consists of four major components: files managing subsystem, drawing subsystem, simulation subsystems and analysis subsystem. We also have shown in an example how the subsystems to work to automatically transform rules into Petri nets and to draw, set, execute and analyze Petri nets.

References

[buga94] A.J. Bugarin and Barro. A reasoning algorithm for high level fuzzy Petri nets," *IEEE Transaction on Fuzzy Systems*, 4(3): 282-294, 1994.

[chen90] S.M. Chen, J.M. Ke, and J.F. Chang. Knowledge representation using fuzzy Petri nets. *IEEE Transactions on Knowledge and Data Engineering*, 2(3):311-319, 1990.

[ivar92] Ivar Jacobson. Object-Oriented Software Engineering, Addison Wesley, 1992.

[kona96] A. Konar and A. K. Mandal. Uncertainty Management in Expert Systems Using Fuzzy Petri Nets, *IEEE Transactions on Knowledge and Data Engineering*, 8(1): 96-105, 1996.

[liu97] K.F.R. Liu, J. Lee, S.J. Yang, and W.L. Chiang. Fuzzy Petri Nets for Modeling Rule-based Reasoning, *In ICTAI'97(to appear)*, 1997.

[loon88] C.G. Looney. Fuzzy Petri nets for rule-based decisionmaking, *IEEE Transactions on Knowledge and Systems, Man, and Cybernetics*, 18(1): 178-183, 1988.

[luck95] D.C. Luckham, J. Kenney, L.M. Augustin, J.

Vera, D. Bryan, and Walter Mann. Specification and Analysis of System Architecture Using Rapide. *IEEEE Transaction on Software Engineering*, 22(4): 336-355, 1995.

[mura89] T. Murata, Petri Nets: Properties, Analysis and Application. *Proceedings of the IEEE*. 77(4): 541-580, 1989.

[scar96a] H. Scarpelli, F. Gomide, and W. Pedrycz. Modeling fuzzy reasoning using high level fuzzy Petri nets. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 4(1): 61-85, 1996 ·

[scar96b] H. Scarpelli, F. Gomide, and R. Yager. A reasoning algorithm for high level fuzzy Petri nets. *IEEE Transaction on Fuzzy Systems*, 4(3):282-294, 1996 ·

[yang97] S.J. Yang, J. Lee, W.C. Chu and W.T. Huang. A fuzzy Petri nets based mechanism for fuzzy rules reasoning. *In COMPSAC'97(to appear)*, 1997 .

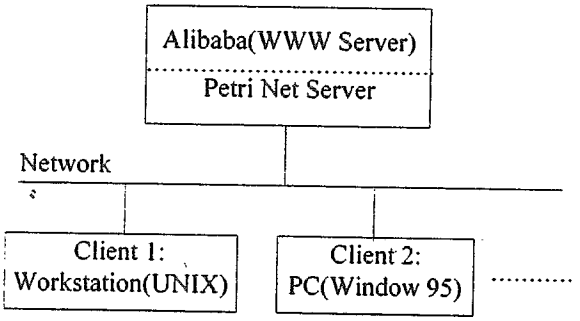

Figure 1. Architecture of network for PNMAS.

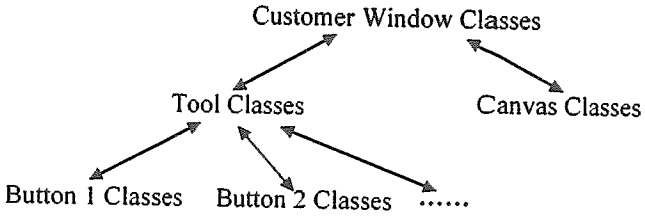

Figure 5. Customer window classes

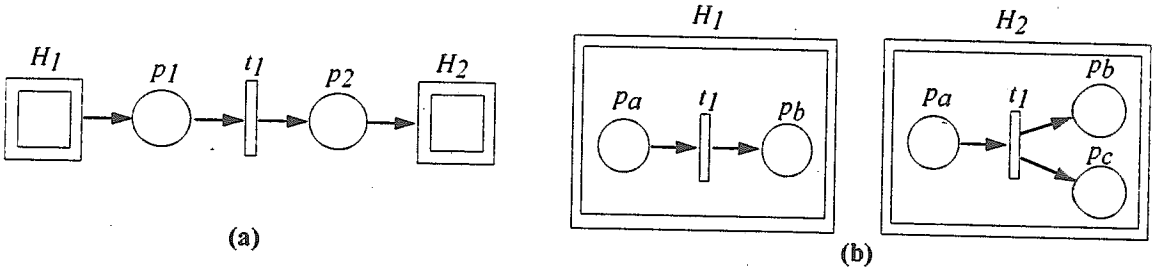

Figure 2. (a) Petri net with hierarchies, and (b) the structures of hierarchies.

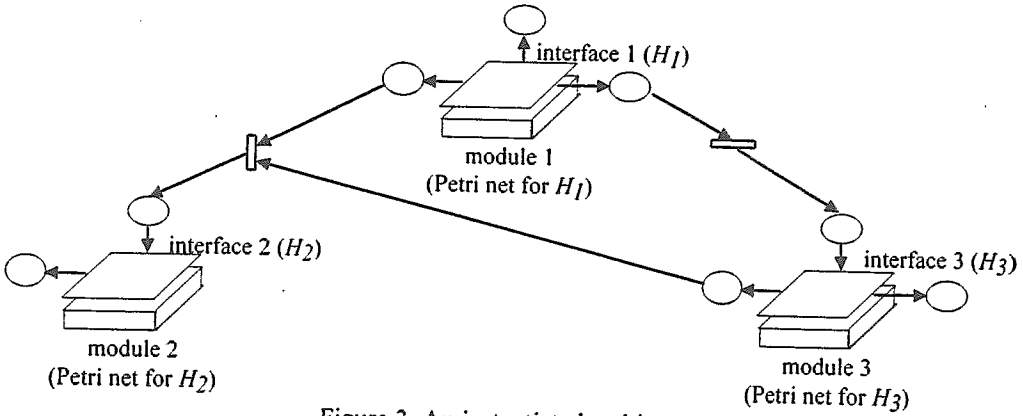

Figure 3. An instantiated architecture.

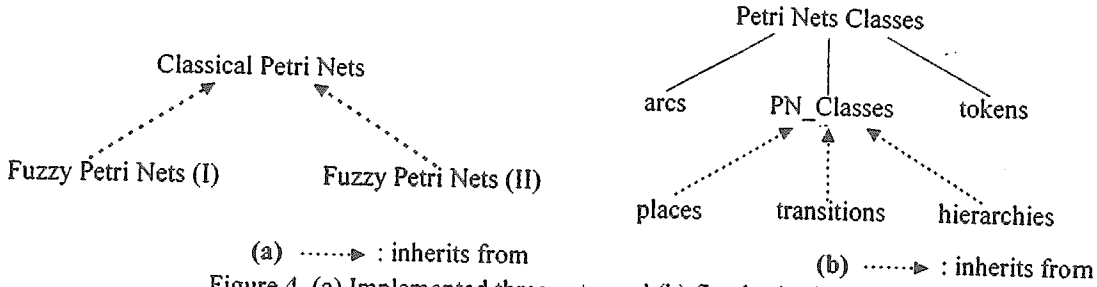

Figure 4. (a) Implemented three nets, and (b) five basic classes.

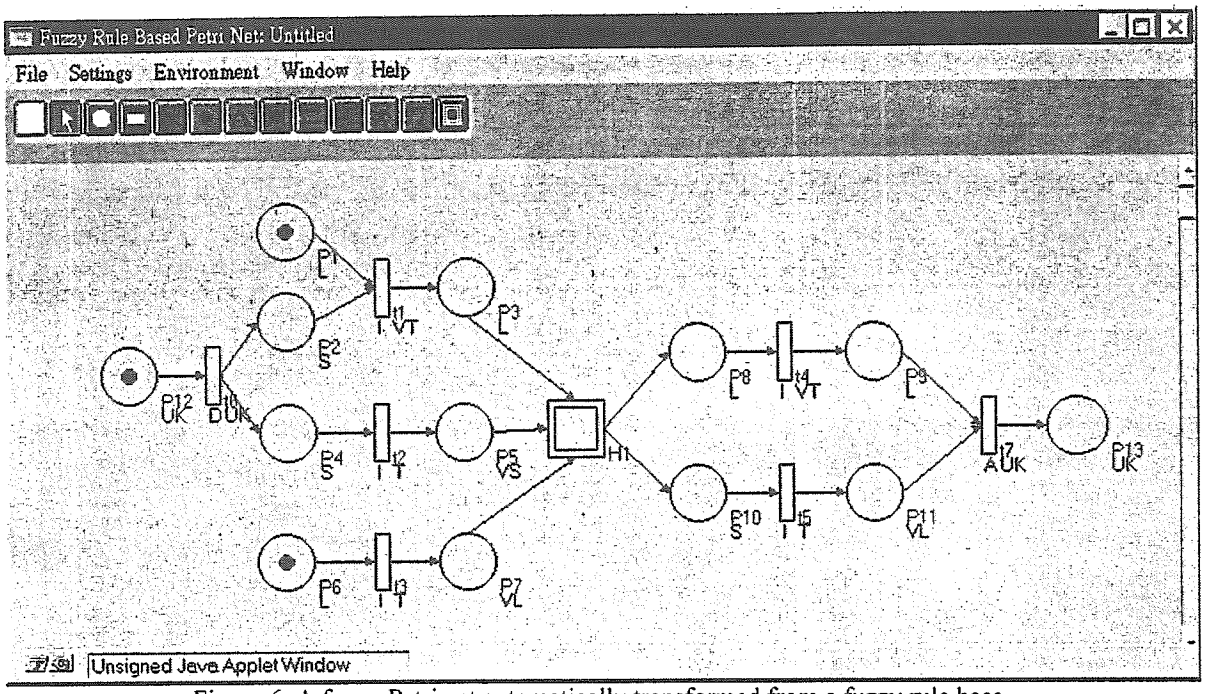

Figure 6. A fuzzy Petri net automatically transformed from a fuzzy rule base.

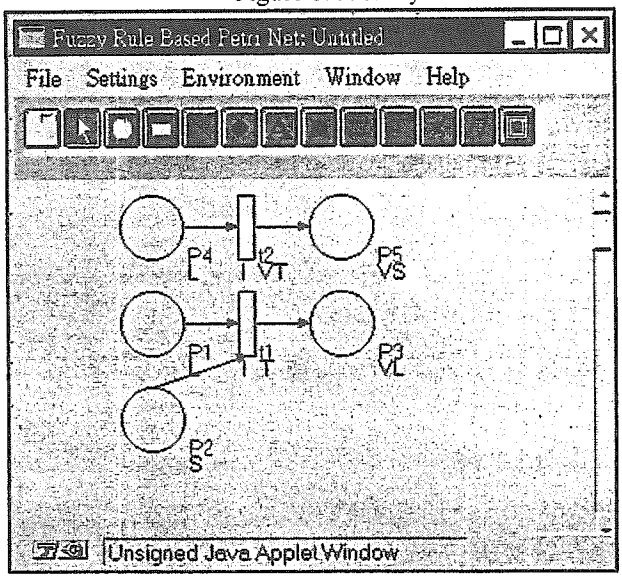

Figure 7. A fuzzy Petri net represented by hierarchy H1

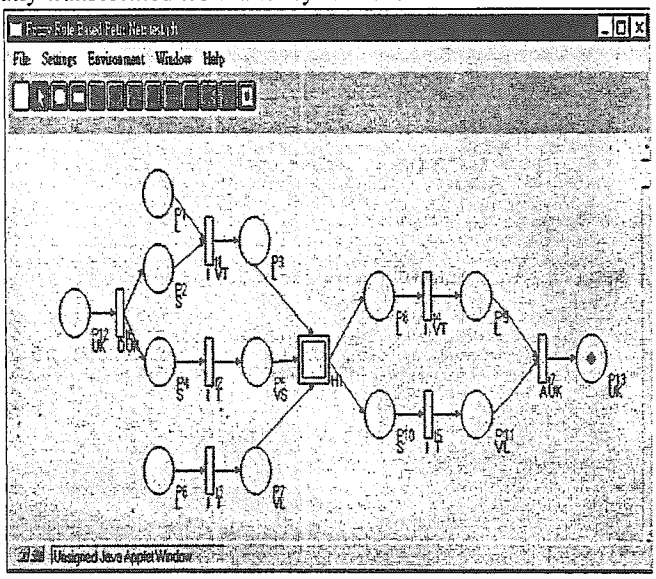

Figure 8. The result of execution.

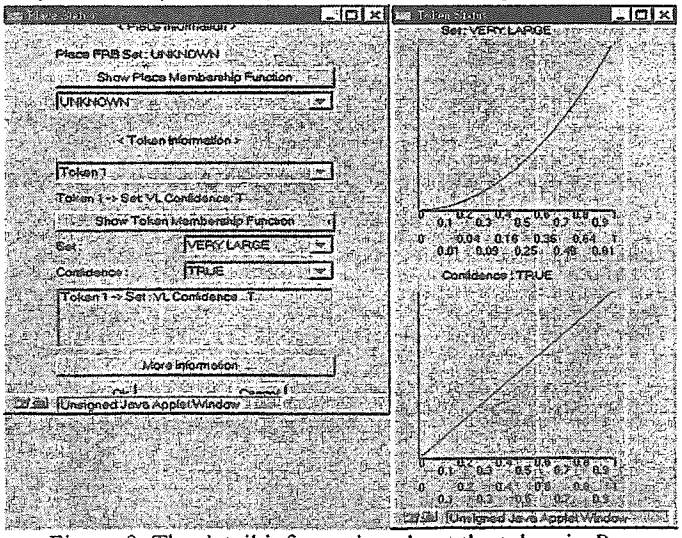

Figure 9. The detail information about the token in P_{13}.