



Knowledge Representation of Coloured Petri Nets

* A.Kumaravel, ** M.Merlin Mexa

* Department of Computer Science and Engineering. Bharath University, Selaiyur, Chennai-600073, India

** Department of Computer Science and Engineering. Bharath University, Selaiyur, Chennai-600073, India

ABSTRACT

Colored Petri Nets (CP-nets or CPN) is a graphical oriented language for design, specification, simulation and verification of systems. Petri nets provide the primitives for the description of the synchronization of concurrent processes, while programming languages provide the primitives for the definition of data types and the manipulation of data values. The CPN language makes it possible to organize a model as a set of modules, and it includes a time concept for representing the time taken to execute events in the modeled system. In this paper we consider the student dataset with multi classes and propose the classification for each type of data, and the rules are generated for the dataset. These rules are applied to the CPN Tools for the creation of Colored Petri net.

Keywords : Coloured Petri Nets, Related works, Classification, Analysis, Examples

I. INTRODUCTION

CPN Tools is a tool for editing, simulating and analyzing un-timed and timed, hierarchical Petri nets (CPN or CP-nets)[1]. CPN Tools is intended to replace Design/CPN [2], which is a widespread software package for CP-nets. In addition to Design/CPN, CPN Tools can be compared to other Petri net tools such as ExSpect, GreatSPN, and Renew which are all described in the Petri Nets Tool Database [3].

Design/CPN was first released in 1989 with support for editing and simulating CP-nets. Since then a significant amount of time has been invested in developing efficient and advanced support both for simulation and for generating and analyzing full, partial, and reduced state spaces. While the analysis components of Design/CPN have steadily improved since 1989, the graphical user interface has remained virtually unchanged.

CPN is the Combination of Petri Nets and Programming Language: Control structures, synchronization, communication, and resource sharing are described by Petri Nets, Data and data manipulations are described by functional programming language. Colored Petri Nets is developed at University of Aarhus, Denmark over the last 25 years. The functionality of the CPN Tools and Design/CPN is the same:

1. Editing and syntax check of CP-nets.
2. Interactive and automatic simulation.
3. Construction and analysis of state spaces.
4. Communication with other tools.
5. Simulation based performance analysis.
6. Graphical animation of simulation results.

A CPN model of a system describes the states of the system and the events (transitions) that can cause the System to change state. By making simulations of the CPN model, it is possible to investigate different scenarios and explore the behaviors of the system. Very often, the goal of simulation is to debug and investigate the system design. CP-nets can be simulated interactively or automatically. With Petri Nets (CP-nets) it is possible to use data types and complex data

manipulation. Each token has attached a data value called the token colour. The token colours can be investigated and modified by the occurring transitions [4].

The practical application of CPN modelling and analysis relies heavily on the existence of computer tools supporting the creation and manipulation of models. CPN Tools [5] is a tool suite for editing, simulation, state space analysis, and performance analysis of CPN models. The user of CPN Tools works directly on the graphical representation of the CPN model.

The graphical user interface (GUI) of CPN Tools has no conventional menu bars and pull-down menus, but is based on interaction techniques such as tool palettes and marking menus. Licenses for CPN Tools can be obtained free of charge via the CPN Tools web pages [5]. CPN Tools is currently licensed to more than 4,000 users in more than 115 different countries and is available for MSWindows and Linux.

Coloured PetriNets (CP-nets or CPNs) [9]-[11] is a graphical language for constructing models of concurrent systems and analyzing their properties. CP-nets is a discrete-event modelling language combining Petri nets [13] and the functional programming language CPN ML which is based on Standard ML [14],[15]. The CPN modelling language is a general purpose modelling language, i.e., it is not focused on modelling a specific class of systems, but aimed towards a very broad class of systems that can be characterized as concurrent systems. Typical application domains of CP-nets are communication protocols [6], data networks [5], distributed algorithms and embedded systems. CP-nets are, however, also applicable more generally for modeling systems where concurrency and communication are key characteristics. Examples of these are business process and workflow modelling, manufacturing systems [11], and agent systems. Examples of industrial applications of CP-nets within different domains are available via [12]. An introduction to the practical use of CP-nets is also given in [10], [12].

The section 1 consists of Introduction, section 2 includes the project related works, section 3 contains the detailed information about the Coloured Petri Net, the analysis includes in section 4 and section 5 explains the conclusion.

II. RELATED WORKS

In this paper I created the student dataset consists of different instances and attributes and I classify the dataset by choosing the JRIP and creating the cross-validation for the dataset, the run information will displayed in the classifier output. It consists of the Relation, how many instances, how many attributes and the JRIP rules, Time taken to build a model, Stratified cross-validation, Detailed Accuracy by class and confusion Matrix. Consider the Student dataset & Process Flow Diagram:



Fig. 1 Student Dataset

The Student dataset consists of the attributes such as Student Name, Course Name, Marks and Performance. Using the data set the JRIP rules are created and these rules are applied in the Coloured Petri Net

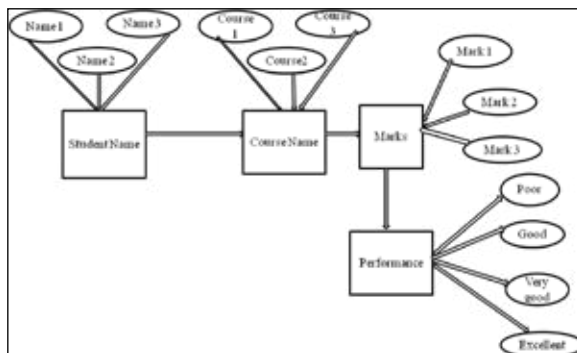


Fig.2 Process Flow Diagram

In Fig. 3, the Process Flow diagram consists of the places for each attributes, the student name is fired to the transition and the course name is send to the transition, after the marks received according to the marks the performance will be fired.

III. COLOURED PETRI NET

A Coloured Petri Net (CPN) [10] is a tuple $CPN = (\Sigma, P, T, A, N, C, G, E, I)$ where

- 1) Σ is a finite set of types called *Colour sets*, which are finite and non – empty. The set of colour sets determines the types, operators and functions that can be used in the net inscriptions.
- 2) P is a finite set of *places*
- 3) T is finite set of *transitions*.
- 4) A is a finite set of arcs such that $P \cap T = P \cap A = T \cap A = \emptyset$
- 5) $N: A \rightarrow P \times T \cup T \times P$ is a node function
- 6) $C: \rightarrow \Sigma$ is a *colour function*
- 7) G is a *guard function*, It is defined from T into expres-

sions such that : $\forall t \in T: [Type(G(t)) = Bool \wedge Type(Var(G(t))) \subseteq \Sigma]$
 8) E is an arc expression function, It is defined from A into expressions such that :
 $\forall a \in A: [Type(E(a)) = C(p(a))_{MS} \wedge Type(Var(E(a))) \subseteq \Sigma]$

Where $p(a)$ is the place of $N(a)$ and C_{MS} denotes the set of all multisets over C

- 9) I is an initialization function, I is defined from P into expressions such that:
 $\forall p \in P: [Type(I(p)) = C(p)_{MS} \wedge Var(I(p)) = \emptyset]$

A small example of a CP-net is shown in

Fig. 1 It consists of the places and transitions, The place $p1$ and $p2$ having the token value as string. These token value is fires through the transition $t1$ and the output is displayed on the place $p3$.

The ellipses and circles are called *places*. They describe the states of the system. The rectangles are called *transitions*. They describe the actions. The arrows are called *arcs*. The arc expressions describe how the state of the CP-net changes when the transitions occur. Each place contains a set of markers called *tokens*. In contrast to low-level Petri nets (such as Place/Transition Nets), each of these tokens carries a data value, which belongs to a given *type*. Coloured Petri Nets have got their name because they allow the use of tokens that carry data values and can hence be distinguished from each other – in contrast to the tokens of low-level Petri nets, which by convention are drawn as black, “uncoloured” dots.

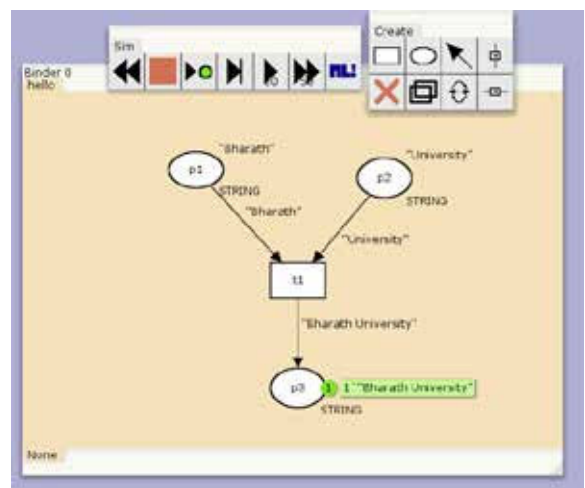


Fig.3 Simple Example

The Fig.3 shows the simple example of Coloured Petri Net. It consists of two places $p1$ and $p2$ with place type string and the initial marking. The two places are fired on the transition $t1$ and we will get the output in the place $p3$. Create Tool is used to create the place, transition, arcs etc. The Simulation Tool consists of Goes to the initial state tool and Executes the transition tool etc.

IV. ANALYSIS OF COLOURED PETRI NETS

During the construction of a CP-net, simulations are used to validate the CPN model, i.e., to check that it has the expected behavior. It is customary to work in an iterative way. In the early phases, the CPN model is simple, covering only selected parts of the system and ignoring many aspects of the final system. Later the scope of the model is extended and more details are added. By making simulations during the entire design process, and not just at the very end, the modelers learn about the system – in a similar way as when prototyping is used. This means that design errors can be removed at an early stage and it also means that the designers acquire new knowledge about the system – knowledge

that can be used in the remaining parts of the design process [4].

When a CPN model has been debugged, it can be analyzed in different ways. First of all, it is possible to use automatic simulations. They are similar to program executions and can be very fast with several hundreds/thousands occurring transitions per second. It is possible to specify time delays that describe the duration of the different actions in the modelled system. In this way we can make simulations that investigate the performance of the system.

As for all other formal languages, the practical use of CP-nets is highly dependent on the existence of adequate tool support. For this purpose, we have developed the Design/CPN tool package supporting the construction, editing, syntax check, and simulation of large, modular CP-nets, with or without time delays. Design/CPN also supports construction and analysis of state spaces, allowing the user to verify a large variety of different behavioral properties. The tool package is distributed free of charge to all kinds of users (including commercial companies) [4].

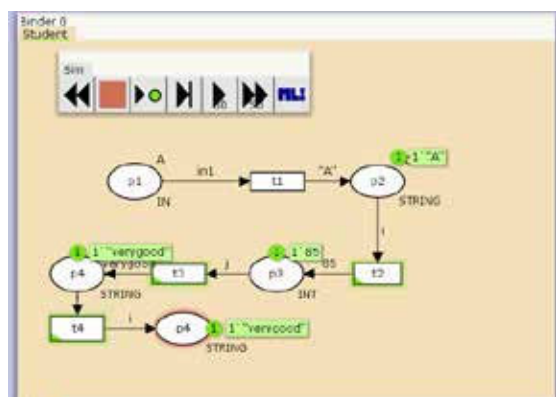


Fig. 4 Student Information Example

The CPN Tools consists of four places and four transitions, the place p1 consists of the token value and these token values are fired on t1 transition. Similar way the p2 will also fires on t2 transition; finally the output will displayed on the t4 transition.

V. CONCLUSIONS

This paper uses the CPN Tools to analyze and validate the dataset. The dataset consists of attributes and instances, it is classified by the JRIP to evaluate the rules and it will be viewed in the classifier output. These rules are applied to the Coloured Petri net Tool and the tokens are fired to the transition, the output token is placed on the end place.

ACKNOWLEDGEMENTS

This work was supported by the management of Bharath University, I would like to thank for the encouragement of this research work.

REFERENCES

- [1] Anne Vinter Ratzer, LisaWells, Henry Michael Lassen, Mads Laursen, Jacob Frank Qvortrup, Martin Stig Stissing, MichaelWestergaard, Søren Christensen, and Kurt Jensen: CPN Tools for Editing, Simulating, and Analyzing Coloured Petri Nets. W.M.P. van der Aalst and E. Best (Eds.): ICATPN 2003, LNCS 2679, pp. 450–462, 2003. c Springer-Verlag Berlin Heidelberg 2003 | [2] Design/CPN. Online: <http://www.daimi.au.dk/designCPN/>. | [3] Petri Nets Tool Database. Online: <http://www.daimi.au.dk/PetriNets/tools/db.html>. | [4] Kurt Jensen: Coloured Petri Nets, URL: <http://www.daimi.aau.dk/~kjensen> | [5] CPN Tools.: <http://www.daimi.au.dk/CPNTools/> | [6] Billington, J., Diaz, M., Rozenberg, G. (eds.): Application of Petri Nets to Communication Networks, vol. 1605. Springer, Berlin (1999) | [7] Billington, J., Gallasch, G.E., Han, B.: A Coloured Petri Net approach to protocol verification. In: Desel, J., Reisig, W., Rozenberg, G. (eds.) Lectures on Concurrency and PetriNets. Advances in Petri Nets. In: Proceedings of 4th Advanced Course on Petri Nets, Lecture Notes in Computer Science, vol. 3018 pp. 210–290. Springer, Berlin (2004) | [8] Desrochers, A.A., Al-Jaar, R.Y.: Applications of Petri Nets in Manufacturing Systems: Modeling, Control, and Performance Analysis. IEEE, (1994) | [9] Jensen, K.: Coloured Petri Nets. Basic concepts, analysis methods and practical use. Basic Concepts, vol. 1. Springer, Berlin (1992) | [10] Jensen, K.: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Practical use, vol. 3. Springer, Berlin (1997) | [11] Kristensen, L.M., Christensen, S., Jensen, K.: The Practitioner's Guide to Coloured Petri Nets. Int. J. Softw. Tools Technol. Transf. 2(2), 98–132 (1998) | [12] Kristensen, L.M., Jørgensen, J.B., Jensen, K.: Application of Coloured Petri Nets in System Development. In: Lectures on Concurrency and PetriNets. Advances in Petri Nets. Proceedings of 4th Advanced Course on Petri Nets. Lecture Notes in Computer Science, vol. 3098, pp. 626–685. Springer, Berlin (2004) | [13] Reisig, W.: Elements of Distributed Algorithms: Modeling and Analysis with Petri Nets. Springer, Berlin (1998) | [14] Standard ML of New Jersey. <http://www.smlnj.org> | [15] Ullman, J.D.: Elements of ML Programming. Prentice-Hall, Englewood Cliffs (1998) | | |