Available online at http://acfa.apeejay.edu

APEEJAY JOURNAL OF COMPUTER SCIENCE AND APPLICATIONS

ISSN:0974-5742(P)
Vol. 2, January 2014

Advanced Petri Nets: A Review

Er.Kawalpreet Singh1 Er.Rajandeep Singh 2

1Assistant Proff., NWIET, Moga (Punjab)

2Assistant Proff., Department of Electronics Technology GNDU Amritsar.

kawal.engg@gmail.com[1],rajandeep1987@gmail.com[2]

Abstract

Petri net is one of the most popular graphical and mathematical tools available for studying systems of discrete event nature, due to its ability to model the precedence relations and structural interactions of random, concurrent, asynchronous events. Petri nets have been widely used in the modeling, control and performance analysis of such systems.

1. Introduction

Petri nets have broad application areas such as robotic tasks and artificial intelligence. Petri nets provide a uniform environment for modeling, formal analysis, and design of discrete event systems [1, 2]. One of the major advantages of using Petri net models is that the same model is used for the analysis of behavioural properties and performance evaluation, as well as for systematic construction of discrete event simulators and controllers.[3] Petri net (PN) is a modeling tool for describing and analyzing discrete-event dynamic systems, and could find applications in many areas such as computer systems, manufacturing systems and power systems. The important difference between the complex Petri net formulation and ordinary PNs is the range of the input functions, output functions, and initial marking. As in the case of the ordinary PN, a graphical representation presents this information in a more accessible manner.[4]

2. Petri Net

The major research aim of a PN system is the organizational structures and dynamic behaviors in a system. It focuses on various changes that may happen in the system and their relationships. A triple (S, T; F) is called a net, if and only if it satisfies the following conditions:

(1) $S \cap T = \Phi$

(2) $S \cup T \neq \Phi$

(3) $F \subseteq (S \times T) \cup (T \times S)$

(4) $dom(F) \cup cod(F) = S \cup T$

with $dom(F)=\{x| \exists y: (x,y) \in F\}$, $cod(F)=\{y| \exists x: (x,y) \in F\}$ Where, S and T are finite nonempty sets of states and transitions, respectively; F is the flow relationship between S and T; “×” is the Cartesian product; dom(F) and cod(F), respectively called the domain and codomain of F, could be interpreted as the first and the last element of each individual set of F [5]. A PN includes places, transitions, arcs and tokens. A simple PN model is shown in Figure 1. In the figure, each cycle “○” represents one place, P; the vertical line “│” represents one transition, T; each arc “→” represents a flow relationship, F; and each dot “•” in the place represents one token.

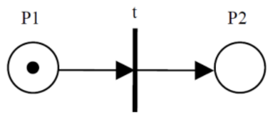

Figure 1. A PN Model

The building blocks of a Petri net are places, transitions, arcs and tokens. Places may contain tokens, which represent the value of the condition or object. Transitions model activities, which change the value of conditions or objects.

3. Advanced Petri Nets

If the basic PN model as described before is employed to model a large-scale power system, the size of the PN model is exponentially increased with that of the power system. In addition to the conventional simplifying measures such as layering and the abstract synthesis of modules, replacing the basic PN by an advanced one is also a choice. To describe a system better, a basic PN could be extended to an advanced PN, such as the predicate/transition net, colored PN (CPN), object-oriented PN (OPN), fuzzy PN (FPN), hybrid PN (HPN), PN with changeable structure (PNCS), timed PN (TPN), stochastic PN (SPN), generalized stochastic PN (GSPN), timed CPN (TCPN), stochastic CPN (SCPN) and timed OPN (TOPN).

4. Predicate/transition net

The transmission medium is modeled in the middle and the two processes, send message/receive ack and send ack/receive message, are on either side. Because Petri nets are fairly simple structures, the representation of larger systems can get clumsy and the proof of such nets gets very involved. Communication protocols have a large number of states and it is difficult to capture all of the interactions using simple Petri nets. Predicate/Transitions nets (PrT) are a more generalized version of Petri nets [6,7].

A PrT net is a tuple (P, T, F, Σ, L, ϕ, M0) [8], where:
• P is a finite set of predicates (first order places), T is a finite set of transitions ($P \cap T = \emptyset$, $P \cup T \neq \emptyset$), and $F \subseteq (P\times T) \cup (T\times P)$ is a set of arcs. (P, T, F) forms a directed net.
• Σ is a structure consisting of some sorts of individuals (constants) together with some operations and relations.
• L is a labeling function on arcs.
• ϕ is a mapping from a set of inscription formulae to transitions
• M0 is the initial or current marking.

PrT nets are a high-level formalism of Petri nets. Parameterized reachability trees [9] exploit parameterized markings as a means for folding reachability trees of PrT nets so that a number of concrete states can be condensed into a generic state.

5. Colored PN (CPN)

CP-nets have been developed from Predicate/Transition Nets. It gives Combination of Petri Nets and Programming Language. The control structures, synchronization, communication, and resource sharing are described by Petri Nets. Data and data manipulations are described by functional programming language. An ordinary Petri net (PT-net) has no types and no modules. Only one kind of tokens and the net is flat. But in this, it is possible to use data types and complex data manipulation. Each token has attached a data value called the token colour[10]. The token colours can be investigated and modified by the occurring transitions. If the CPN has infinite types, such as the integers, text strings or reals, the equivalent Petri Net may become infinite. The main characteristics of CP-nets is combination of text and graphics, declarations and net inscriptions are specified by means of a formal language, e.g., a programming language.

A Coloured Petri Net is a tuple CPN = (S, P, T, A, N, C, G, E, I) satisfying the following requirements:
(i) S is a finite set of non-empty types, called colour sets.
(ii) P is a finite set of places.
(iii) T is a finite set of transitions.
(iv) A is a finite set of arcs such that:
P Ç T = P Ç A = T Ç A = Ø.
(v) N is a node function. It is defined from A into P ´ T È T ´ P.
(vi) C is a colour function. It is defined from P into S.
(vii) G is a guard function. It is defined from T into expressions such that:

The set of colours is finite. Colors can be modified during transition firings, and the same transition can perform diffcrcnt transformations for tokens of different colours. Colours can thus distinguish tokens, and this allows to “fold” similar subnets of a net into a single subnet, reducing the model complexity. In coloured nets, this associated information is called a “colour” of a token. Token colours can be quite complex, for example, they can describe the contents of a message package or the contents of a database. Colored Petri nets can he defined as an extension of marked Petri nets, as in [11].

6. Object-oriented PN (OPN)

The term object-oriented is generally used to describe a system that deals primarily with different types of objects, and where the actions one can take depend on what type of object are manipulated. The methods are

based on simple mathematical models of abstraction and classification. Petri Net has at least, two-object-oriented extensions (a). LOOPN (Language for OO Petri-Nets) and LOOPN++, (b) CO-OPN (Concurrent OO Petri-Nets) and CO-OPN/2. As object orientation was adopted for programming languages, extension to OO-nets inspired from object-oriented programming is a natural flow. The main characteristic of OO-nets is that a net can be in another net as a token[12].
The usual trade-off between modeling power and analytical tractability also applies to hybrid PN models[17].

7.Fuzzy PN (FPN)

With the growth in the complexity of modern industrial, and communication systems, PN found themselves inadequate to address the problems of uncertainty, and imprecision in data. This gave rise to amalgamation of Fuzzy logic with Petri nets and a new tool emerged with the name of Fuzzy Petri Nets (FPN). The numbers of ways have been proposed for combining PN with fuzzy logic, according to different applications. But with the increasing applications of these nets, there is an increase in the ambiguity about their types and structures. As PN can be timed and/or colored, similarly FPN can also be timed and/or colored to include the temporal effect and/or enhance their visibility. PN are graphical, and mathematical modeling tool usually used for discrete event systems. FPN was being proposed as 8-tuple in [13].

$FPN=(P,T,D,I,O,f,\alpha,\beta)$
Basic steps involved in the modeling of FPN are; first extract information from experts and/or databases, and then fuzzify the information from crisp set to fuzzy set by defining membership function. Next step is to construct PN by designing rule base, and inference rules, and in the last, defuzzifiation of the results. From industrial engineering point of view, always there are demand fluctuations and machine breakdown. Both fluctuation and breakdown can be better modeled by FPN to simulate real world processes. On the basis of structures and algorithms FPNs have been classified as; Basic Fuzzy Petri Nets (BFPN), Fuzzy Timed Petri Nets (FTPN), Fuzzy Colored Petri Nets (FCPN), Adaptive Fuzzy Petri Nets (AFPN), and Composite Fuzzy Petri Nets (CFPN).

8. Timed PN (TPN)

Timed Petri Nets (TPN) were found useful for performance evaluation of systems in general and manufacturing systems in particular but have been criticized as becoming too large, cumbersome and time consuming when large numbers of products (entity types) and work stations (resource types) are involved. The two main extensions of Petri Nets with time are Time Petri Nets (TPNs) and Timed Petri Nets. For TPNs, a transition can fire within a time interval whereas for Timed Petri Nets it fires as soon as possible. Among Timed Petri Nets, time can be considered relative to places or transitions. The two corresponding subclasses namely P-Timed Petri Nets and T-Timed Petri Nets are expressively equivalent. TPNs form a subclass of Time Stream Petri Nets which were introduced to model multimedia applications. The TPN approach puts into evidence the flow realization along a variety of network paths and may consider different network structures.[14]

We add time to PNs by introducing the notion of temporal constraints on transitions. We add a lower bound d and an upper bound D to transitions: if transition t is associated with constraint [d, D] (with d £ D), then, after t is enabled, it must fire no less than d and no more than D time units after it is enabled (unless it is disabled before). Two timed Petri nets are said to be equivalent, if they have the same topological structures, i e, the same.
$N = (P,T,A,B)$
Starting with the same initial marking (any initial marking), the two nets will give the same token distribution at any time T. if when multiple transition are enabled, they follow the same sequence of tiring these transitions. First, it can be seen that conventional Petri nets are special cases of timed Petri nets.

9. Stochastic PN (SPN)

Stochastic Petri Nets are a formalism developed in the field of computer science for modeling system performance. Software has been written to implement and solve models defined using SPN formalism. SPNs consist of places and transitions as well as a number of functions. The initial state of the system is represented by the initial marking. SPNs can be represented graphically, with places represented as circles and transitions as rectangles, and input and output functions as directed arcs. [15] The properties of SPNs are

SPNs have discrete state spaces, defined by the number of objects in each place (the marking). Places can be linked to transitions as input places, and transitions can be linked to output places. Transitions are said to be enabled when there are enough objects in each of the input places. Enabled transitions can fire, removing objects from their input places and adding objects to their output places. Enabled transitions fire according to exponential distributions, characteristic of Markov Processes. Stochastic Petri nets techniques are attractive because they provide a performance evaluation approach based on formal description.
An SPN is defined as a 7-tuple
SPN= (P, T, I(.), O(.), H(.), W(.), M0)
Where, PN = (P, T, I(.), O(.), H(.), M0) is the P/T system underlying the SPN

10. Generalized stochastic PN (GSPN)

The class of Petri nets obtained by eliminating timing from generalized stochastic Petri net (GSPN) models while preserving the qualitative behavior is identified. Generalized Stochastic PNs (GSPNs) have two different classes of transitions: immediate transitions and timed transitions. Once enabled, immediate transitions fire in zero time. Timed transitions fire after a random, exponentially distributed enabling time as in the case of SPNs. For timed transitions, the firing rate is, by default, marking independent, but the user can select a marking-dependent operation (the same way as for SPNs). [16] The simulation procedure is similar to the SPN case, the only difference occurring in the case of the immediate transitions that fire first; priorities/probabilities can be associated to these transitions, in order to resolve the conflicts. A GSPN is defined as an 8-tuple:

SPN= (P, T, pri(.), I(.), O(.), H(.), W(.), M0)

Where, PN = (P, T, pri(.), I(.), O(.), H(.), M0) is the P/T system with priority underlying the SPN. The transitions have an exponentially distributed delay.

11. Petri Nets with in-changeable structure

Petri Nets with in-changeable structure (PN-iCS) PN - iCS = LN,$PN_0{}^*$,H) , where LN is the global structure of system, $PN_0{}^*$ is a PN system, and H is a map from T^0 ($T^0 \subseteq T$)to LN .
LN = (P,T, F)is a domain of the map H . P , T and F appear as global place set, transition set and arc set, respectively.
$PN^*{}_0 =(P^*,T^*, F^*,\mu^{00})$ $PN^*{}_0 \subseteq PN$) is the initial structure and state of system, where
$S^* \subseteq S$, $T^* \subseteq T$
$F^* \subseteq (S^* \times T^*)\cup(T^*\times S^*) \subseteq F$. μ^{00} is the initial state in the initial structure, μ^{ij} is the change of state., where i is the representative that a transition changing structure is firing, when i changes into i +1 , and j is the representative that a transition is firing, when j changes into j +1.
H : $T^o \rightarrow LN$ ($T^o \subseteq T$) is a map that system structure changes with inter factors.
$\forall t \in T^o$,$H(t) =(O, G_t)(O_t \cap G_t = \emptyset)$, where O_t is the deleting element set, and G t is adding the element set. In $O_{t\,=}$ (O_{Pt} ,O_{Tt} O_{Ft}), O_{Pt} ,O_{Tt} O_{Ft} appear as deleting place set, transition set and arc set, respectively. In, G_t =(G_{Pt}, G_{Tt}, G_{Ft}), G_{Pt}, G_{Tt}, G_{Ft} appear as adding place set, transition set and arc set, respectively. Arcs connected with places and transitions, which are deleted, must be deleted. Places and transitions connected with added arcs must be added. When palces are deleted or added, will delete or add the corresponding dimensions of marking.

12. Petri Nets with Systematized Changeable Structure

By definitions the petri Nets with systematized changeable structure (SPN-SCS)
PN -SCS_k = (PN - iCS_k U_k), $k \in N$. k is the counter that indicates the number of times that the system has been changed by out factor, $k \in N$. PN - iCS_k is a Petri Nets with in-changeable structure.
$U_k = (U^1{}_k\ U^2{}_k\ U^3{}_k)$ is given by the environment or some human requirements after the k th change to the system. $U^1{}_k$ and $U^2{}_k$ appear as deleting and adding set, respectively.
$U^1{}_k\ U^2{}_k$ = . $U^3{}_k$ is the modifying set($\mu_k{}^{mknkmk}$ and H_k) in k PN - iCS . m_k is the total number of times that the system structure has been changed by inter factors under the k^{th} out-changeable structure . $n_k{}^{mk}$is the counter that indicates the number of times that the system state has been changed under the $m_k{}^{th}$ in-changeable structure. [19,20]
$U^1{}_k= (U^1LN_kU^1PN^*{}_{mkk})$, $U^1{}_{Hk}U^1{}_k$ PN- iCS_k,

$U^2{}_k= U^2{}_{LNk}, U^2{}_{PN^*mkk},U^2{}_{Hk})$,

$U^3_k = U^3_{kmknk}, U^3_{Hk}$

The PN-SCS provides a way of combining Petri nets with Complex Networks.

13. Time object oriented petri nets

The purpose of designing timed hierarchical object-oriented Petri net (TOPN) is to aid in the modeling and analysis of real time systems and bridge the gap between the formal treatment of object-oriented Petri nets and temporal reduction approach for the modeling, analysis, and prototyping of complex time critical systems. TOPN model is a variant HOONet representation that corresponds to the class with temporal property in object-oriented paradigm. Like the HOONet, TOPN is composed of four parts: object identification place (OIP) is a unique identifier of a class; internal timed object net (ION) is a net to depict the behaviors (methods) of a class; data dictionary (DD) declares the attributes of a class in TOPN; and static time interval function (SI) binds the temporal knowledge of a class in TOPN. TOPN is a four-tuple: TOPN= P (OIP, ION, DD, SI), where:
1. OIP=(oip, pid, M0, status), oip, pid, M0 and status are the same as those in HOONet.
   - oip is a variable for the unique name of a TOPN.
   - pid is a unique process identifier to distinguish multiple instances of a class,which contains return address.
   - M0 is the function that gives initial token distributions of this specific value to OIP.
   - status is a flag variable to specify the state of OIP.
2. ION is the internal net structure of TOPN to be defined in the following. It is a variant CPN that describes the changes in the values of attributes and the behaviors of methods in TOPN.
3. DD formally defines the variables, token types and functions (methods) just like those in HOONet (Hong & Bae, 2000).
4. SI is a static time interval binding function, SI: $\{OIP\} \rightarrow Q^*$, where Q* is a set of time intervals.

14. Timed coloured nets

In timed coloured nets, the firing of a transition t can be considered as a three-phase event; first, the token colours are removed from t's input places and are transformed into occurrence colours of the firing transitions, the second phase is the firing time period when the occurrence colours remain "within" the transition t, and in the last phase, occurrence colours are transformed into token colours oft's output. The behaviour of a timed coloured net can be described by a sequence of states and state transitions. Any state description of a timed net must take into account the marking of a net as well as the distribution of occurrence colours in firing transitions. A timed coloured net is a triple, T = (N, u , f) where N is a coloured net, N = (P, T, A, C, **t,** w, mo), and u is a choice function which, for each marking m of N, assigns the "choice" probability to each selection function g from the set Sel(m), u : Sel(m) $\rightarrow R^{0,1}$, in such a way, that C g € S e l (n) 4 e) = 1, and f is a firing-rate function which assigns the (nonnegative) rate of exponentially distributed firing times to each colour and each transition of the net, f : T × C $\rightarrow R^+$, where R^+ denotes the set of nonnegative real numbers.

There are many merits of the Petri nets such as the ability to describe systems and represent knowledge, at various levels of detail, the ability to model the dynamic behaviour of systems, thus optimise the systems performance, the ability to represent concurrency, the ability to represent the system graphically and in a precise manner, the structural generality. The main disadvantage of modelling using Petri nets, is the complexity of the resulting net. The net can easily become very large and difficult to analyse after only a few .levels of decomposition, but nevertheless these techniques provide a valuable insight into the operation of the system.

15. Conclusion

Petri nets have many advantages and have wide application area. In this paper we have discussed many advanced Petri nets as a basic PN could be extended to an advanced PN. These Petri nets are useful in different fields such as robotics and artificial intelligence, also in modern industrial, and communication systems.

References

[1] Desrochers, Alan A. and Robert Y. Al-Jaar, Applications of Petri Nets in Manufacturing Systems: Modeling, Control, and Performance Analysis, Piscataway, NJ, IEEE press, 1995.

[2] Zhou, Mengchu and Frank DiCesare, Petri Net Synthesis for Discrete Event Control of Manufacturing Systems, Kluwer Academic Publishers, 1993.

[3] Richard Zurawski and MengChu Zhou," Petri Nets and Industrial Applications: A Tutorial IEEE. TRANSACTIONS ON INDUSTRIAL ELECTRONICS, VOL. 41, NO. 6, DECEMBER 1994

[4] T.J. Deal, "Complex Token Petri Nets", M.S. Thesis, Rensselaer Polytechnic Institute, Troy, NY, August 2001.

[5] C. Y. Yuan, The Theory and Application of Petri Net, Beijing: Publishing House of Electronics Industry, 2005.

[6] H. J. Genrich, "Predicate/Transition Nets" In: K. Jensen and G. Rozenberg (eds.): High-level Petri Nets. Theory and Application. pp 3-43 Springer-Verlag, 1991

[7] H. J. Genrich and K. Lautenbach, "The Analysis of Distributed Systems by Means of Predicate/ Transition-Nets" In: Kahn, G.: Lecture Notes in Computer Science, Vol. 70: Semantics of Concurrent Computation, pages 123-146. Berlin: Springer-Verlag, 1979.

[8] D. Xu, J. Yin, Y. Deng and J. Ding: A Formal Architecture Model for Logical Agent Mobility. IEEE Trans. on Software Engineering. vol. 29, No. 1, pp. 31-45, Jan. 2003

[9] M. Lindqvist, "Parameterized Reachability Trees for Predicate/Transition Nets," Proceedings of the 11th International Conference on Applications and Theory of Petri Nets, Paris, 1990.

[10] Coloured Petri Nets by Kurt Jensen,Computer Science Department, University of Aarhus, Denmark Kurt Jensen. A brief introduction to colored Petri nets. In: Proc. Workshop on the Applicability of Formal Models, 2 June 1998, Aarhus, Denmark, pages 55-58. 1998

[11] W.M. Zuberek, "Performance evaluation using timed colored Petri nets"; protocols"; Proc. Midwest Symp. on Circuits and Systems, pp.779-782, 1990.

[12] Michael Zapf, Armin Heinzl, Techniques for Integrating Petri-Nets and Object-Oriented Concepts, Working Papers in Information Systems, University of Bayreuth, 1999.

[13] Yung-Hsiang Cheng, and Li-An Yang, "A Fuzzy Petri nets approach for railway traffic control in case of abnormality: Evidence from Taiwan railway system," Expert Systems with Applications, vol. 36, pp. 8040- 8048, 2009.

[14] Tony Spiteri Staines, Using a Timed Petri Net (TPN) to Model a Bank ATM, Proceedings of the 13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems (ECBS'06)

[15] A. Blakemore and S.K. Tripathi, "Automated time scale decomposition and analysis of stochastic Petri Nets," In Proc. 5-th Intern. Workshop on Petri Nets and Performance Models, pages 248--257, Toulouse, France, October 1993. IEEE-CS Press.

[16] M. Ajmone-Marsan, G. Balbo, and G. Conte, "A class of generalized stochastic Petri nets for the performance analysis of multiprocessor systems," ACM Transactions on Computer Systems, 2(1), May 1984.

[17] Jun Li, Ming-dong Li, Yong-feng Diao, and Ying Zhang , One Kind of Petri Nets with Systematized Changeable Structure, Chinese Control and Decision Conference, 2009.

[18] Gupta Deepak, Kewal Krishan Nailwal, and Sameer Sharma. "Minimizing Hiring Cost For Three Stage Flowshop Scheduling For A Fixed Sequence Of Jobs.", Apeejay Journal of Computer Science and Applications", Vol. 1, 2013, pp. 33-38.