TWMS J. App. and Eng. Math. V.12, N.2, 2022, pp. 547-555

TIMED COLORED PETRI NET GENERATING ARRAYS

S. VAITHYASUBRAMANIAN[1], D. LALITHA[2], M. I. MARY METILDA[2], §

Abstract. Motivated by Interval Timed Colored Petri Net and two dimensional Array generating Petri nets, this paper defines an Array Generating Timed Colored Petri Net. In this paper time has been associated with the tokens of the net. Time associated with tokens will delay its availability as a resource. The introduction of time as an attribute of the token has an additional control over the firing sequence of the net. The focus of this paper is on the array language generated by such timed colored Petri net model.

Keywords: Array Language, Catenation, Colored token, Timed Petri net, Token Attributes.

AMS Subject Classification: 68Q45, 03D05, 03D99.

1. Introduction

Petri nets are used to model distributed systems [1]. The components of many such systems exhibit concurrency and parallelism. The net modeling complex systems could be extremely large. Tokens generally represent the resources required for activities to take place. All the tokens in a basic Petri net are black dots. In a complex system the resources may have different attributes. It is not possible to represent the various attributes of the resources by black tokens. Hence basic Petri nets are not suitable to model such systems. Since it is essential to have different types of tokens, in Colored Petri Net (CPN), each token carries a value [2, 4, 3]. The functional programming language SML is used to define the data types and to declare variables. Multi-sets can be used in the places of CPN.
Time is an important aspect in all discrete dynamic systems. Timed Petri Net (TPN) was introduced to model such systems [5, 6, 7, 8, 9, 10]. Time and time delay can be introduced in certain activities of the net. In certain models time is associated with the transition. When an enabled transition fires, the tokens are removed from the input places but are retained for some time before they reach the output places. This delay in firing the

[1] PG and Research Department of Mathematics, D.G. Vaishnav College,
(Affiliated to Madras University), Chennai - 600 106, India.
e-mail: discretevs@gmail.com; ORCID: http://orcid.org/0000-0002-1252-902X.
Corresponding author.
[2] Faculty of Science and Humanities, Department of Mathematics, Sathyabama Institute of Science and Technology,Chennai - 600 119, India.
e-mail: lalkrish2007@gmail.com; ORCID: http://orcid.org/0000-0002-9804-9114.
e-mail: metilda81@gmail.com; ORCID: http://orcid.org/0000-0003-4685-2527.
§ Manuscript received: March 18, 2020; accepted: May 11, 2020.
TWMS Journal of Applied and Engineering Mathematics, Vol.12, No.2

transition is called the firing delay. Wong introduced a model in which time is attached with the place [9]. The tokens coming into that place is not available as a resource for some time period. Some authors mix and associate time with both enabling transition and firing of transitions. When time is introduced in certain activities of the net, the type of delay of these activities will also have to be identified. The delays could be fixed, stochastic or chosen from an interval.
In Interval Timed Colored Petri Net (ITCPN) time is attached to tokens as one of its attributes and it is called the timestamp [11, 12, 13]. If a transition has several input places with tokens, then the transition is enabled to fire only when the built in clock reaches the maximum of all the timestamps of the tokens to be removed. The transition with the least enabling time will be fired first. If more than one transition has the same enabling time then any one of them will be non-deterministically chosen to fire.
Transitions can be labeled by characters of an alphabet. Then any sequence of transitions which takes the initial marking to final marking gives a word over the alphabet. Thus a string language can be generated by a Petri net. The language generated can be used in investigating the properties of the net. Motivated by this, Petri net models have been introduced to generate rectangular and hexagonal array languages [14, 15, 16, 17, 18]. In such Petri net models inhibitor arcs can be used to bring in more control in the generation of array languages [16]. A complex Petri net may be required to generate certain array languages. Hence higher level Petri net can be used to reduce the complexity of the net. String generating Petri nets have been motivation for these array generating Petri nets. In this paper we use the concept of time and color in generating arrays. Different types of tokens can be used. But in this paper only rectangular arrays made up of a given set of characters are used as token. Positive integers can be used as tokens to have a count on the number of times a transition fires. Application of these arrays generation can be found in [19, 20].
This paper is organized as follows. The second section gives all the basic definitions of array generating Petri nets. Some of the concepts have been taken from the existing literature. Some of them have been introduced here for the first time. The section also defines an Array Generating Timed Colored Petri Net and explains it with example.

2. Timed Colored Petri Net

Preliminary Concepts: In this section all the basic notations required to understand Array Generating Petri nets are given. Then an array generating timed colored Petri net is defined. ITCPN has been defined for its application in logistics [13]. We first recall the definition of ITCPN. Any token in an ITCPN has four attributes. A token is represented by $< i, p, v, x >$ a distinct positive integer i, the identification of the token. Distinct tokens will have distinct identification. p is the place in which the token resides. v is the color/ value/ data type of the token and x is a non-negative integer which is the timestamp or the time at which the token will be made available as a resource. If the timestamp is zero, then the token is available as soon as it reaches the output place. Other than this time stamp, transitions are associated with an interval of time. This interval specifies the delay in the token reaching the output place. On firing any transition the timestamp of the token coming into the output place will depend on both the timestamp of the token in the input place as well as the interval associated with the transition fired.

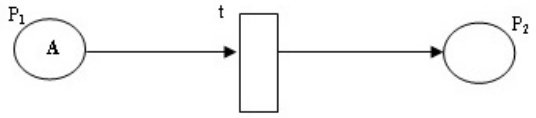

FIGURE 1. Before Firing

2.1. **Notations Used in Array Languages.** Two arrays can be joined column-wise if they have the same number of rows or row-wise if they have the same number of columns. This operation is called catenation. If A and B are two arrays having same number of columns, then $A \otimes B$ denotes the array which is got by joining the two arrays row-wise. If A and B are two arrays having same number of rows, then $A \oplus B$ denotes the array which is got by joining the two arrays column-wise. In formal language theory a_m denotes a column of m a's and a^n denotes a row of n a's. In general if the array A has m rows and n columns, $A \otimes b^n$ denotes joining a row of b's to the array A after its last row and $b^n \otimes A$ denotes joining a row of b's to the array A before its first row. Similarly $A \oplus b_m$ denotes joining a column of b's to the array A, after its last column and $b_m \oplus A$ denotes joining a column of b's to the array A, before its first column.

2.2. **Notations Used in Array Generating Petri Nets.** In array generating Petri nets the effect of firing transitions will depend on the existence of label. All transitions of the net need not be labeled. A transition without label just moves the array in the input place to the output places. Labels are defined as a partial function on T, the set of transitions. $\varphi(t)$ could be a catenation rule. A catenation rule takes the form $A \otimes B$ (or) $A \oplus B$. In this rule the array A is the array coming from the input place. B is an array language. An array language is an infinite set of arrays having a either a fixed number of rows or fixed number of columns. If B is involved in a row catenation then, B will have fixed number of rows but the number of columns of B will take the same value as the number of columns of A. Similarly if B is involved in a column catenation then, B will have fixed number of columns but the number of rows of B will take the same value as the number of rows of A. Example 1 shows how on firing a transition t the array A in the input place is joined with another array B and put in the output place. B is an array language and the number of columns in B will depend on the number of columns of A. So the resultant square array of size 3 is put in the output place. Figure 1 shows the net structure before firing t and figure 2 shows the net structure after firing t. Figure 3 shows the catenation rule associated with the transition and also the arrays involved.

Example 2.1. *Petri net models have been used to generate array languages. Sometimes the basic Petri net could be very complex and hence unwieldy. Using the ideas of ITCPN a new model called Array Generating Timed Color Petri Net [AGTCPN] is defined in this section. This model generates rectangular arrays made over a given alphabet. In this new model, time is added as a component in the attribute of the token. Adding time gives more control on the arrays that gets generated. Every token has four attributes.*

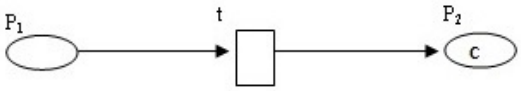

FIGURE 2. After Firing

$$\varphi(t) = A \otimes B$$

$$A = x\ x\ x,\ B = \begin{pmatrix} \bullet \\ \bullet \end{pmatrix}^n \quad C = \begin{matrix} x & x & x \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{matrix}$$

FIGURE 3. Arrays of the net

(i) *Its identity, referred to as its id, a name which is unique for every token*
(ii) *Its position, the place where it resides*
(iii) *Its value, initial value of the token is known*
(iv) *Timestamp, TS, is a non-negative number.*

The attributes of the token takes the form $< id, P, A, TS >$ *. Firing a transition can change any of the attributes of the token it consumes. Its identity, position, value and timestamp could be changed or retained. If the timestamp of the token is 0, then the token will be immediately available on reaching the output place. If the timestamp of the token is 10, then the token will not be immediately available on reaching the output place. Only after 10 units of time it can be used as an input token. A function* $\varphi(t)$ *is defined for every transition t of the net. It defines the changes that are made in the attributes of the tokens consumed. In this model any place of the net carries the same type of token and it is declared along the place. In the net diagram, the label of each place shows the color of the token it can hold. First a sample net is given to explain how the tokens work with time.*

Example 2.2. *This example is given to show how the time component is used in releasing tokens. The colors involved in this sample are two dimensional arrays.* $P = \{P_1, P_2\}, T = \{t_1\}, \sum = \{a, b\}$ *the arrays A and B are placed as the initial marking in* P_1*. TA is the token attributes.*

$$TA = \{< A, P_1, A, 0 >, < B, P_1, B, 5 >\}$$

$$\varphi_{t_1}(< A, P_1, A, 0 >, < B, P_1, B, 5 >) = [< C, P_2, A \otimes b_m, : C, 5 >] \, if \, id =' A'$$

$$\varphi_{t_1}(< B, P_3, B, 5 >) = [< D, P_2, B \otimes a^n, : D, 0 >]$$

$$A = \begin{matrix} a & a \\ a & a \end{matrix},\ B = \begin{matrix} b & b & b \\ b & b & b \end{matrix},\ C = \begin{matrix} a & a \\ a & a \end{matrix},\ D = \begin{matrix} b & b & b \\ b & b & b \\ a & a & a \end{matrix}$$

In this sample P_1 *has two arrays A and B as tokens. Token A is immediately available but B is available only after 5 units of time. Since the transition* t_1 *is ready to fire whenever*

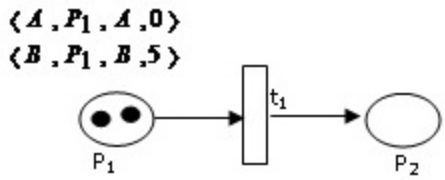

FIGURE 4. Tokens With Time

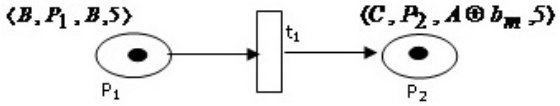

FIGURE 5. Token with lesser timestamp released first

a token becomes available in P_1, t_1 fires. A column of $b's$ is added to the array A and put in P_2. This catenation takes place only if the id of the token which is consumed is 'A'. The new array is stored as C with id 'C' and the timestamp of this token is 5. This means that the token C is available for firing only after 5 units of time. When the built in clock reaches 5 units of time, the token B can be used for firing t_1 again. The token with id A has already been used up before 5 units of time. Hence only one token is left for firing t_1. Thus B has to be used in the second round of firing t_1. When t_1 fires second time a row of $a's$ is added to the array B and put in P_2. This array is stored as D with id 'D' and has no time delay. The position of token before t_1 fires is shown in figure 4. The result of firing the transition t_1 first time is shown in figure 5. The advantage of using the attributes of the token can be clearly seen in example 2. In the basic Petri net, when two tokens are available for firing, one of them will be chosen non-deterministically. Even in a colored Petri net, if many tokens are available in a place, the token will be removed non-deterministically. The output on firing a transition, can be controlled by using the guard functions. The model introduced in this paper has effectively used the timestamps to control the token to be used in order. The array generating timed colored Petri net model is now defined and it is also explained with an example. The timestamp has been used effectively in getting a control over the arrays generated.

Definition 2.1. *An AGTCPN to generate arrays is defined as a nine tuple (C, P, $\sum$, T, I, O, TA, ϕ, F), where C the color set, the data type, P is the set of places of the net, $\sum$ is a subset of printable characters and the arrays will be made of elements from $\sum$, T is the set of transitions such that $P \bigcap T$ is empty, I is the input function from T to bags of places, O is the output function from T to set of places, TA is the initial Token Attributes which is a four tuple of the form $< i, p, v, x >$ where i is the identity of the token, p the place wherein the token lies, v the value, which is the main attribute of the token and x is*

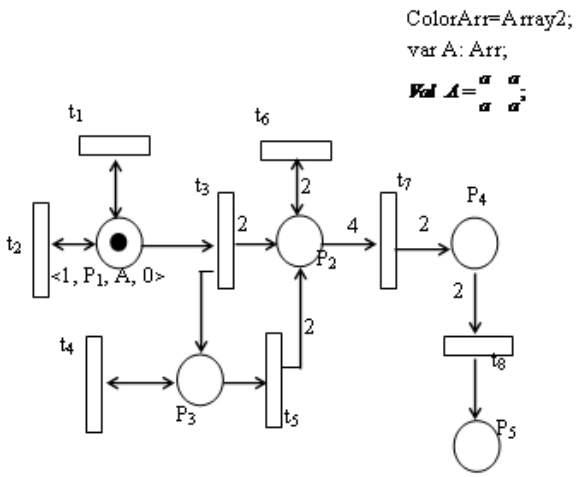

FIGURE 6. Timed Petri Net

the timestamp, ϕ is a function, defined as follows: for all $t \in T$, ϕ_t is a partial function from a set of attributes to a set of attributes. It actually describes the attributes of the tokens that are put in the output place, when the transition t fires and F is a subset of P, the final set of places.

Definition 2.2. *The array language generated by AGTCPN is the set of all arrays that reach the final set of places. This language is denoted by Timed Colored Petri Net Array Language [TCPNAL].*

Example 2.3. *The tokens used in this example are all two dimensional arrays. Every token is a rectangular array made of the alphabet $\sum$. Since all the places can carry only arrays, the data type of the token is not declared along with the name of the place. The language generated by the net in figure 6, can also be generated by an ordinary Petri net. But it would be a very large Petri net and hence would be unwieldy. As far as our model is concerned we used $\sum = \{a, b\}$ which can be extended for different input alphabets and of any size.*
$AGTCPN1 = (C, P, \sum, T, I, O, TA, \phi, F)$ where C is two dimensional arrays, $P = \{P_1, P_2, P_3, P_4, P_5\}$, $T = \{t_1, t_2, ..., t_8\}$, $\sum = \{a, b\}$. The array A defined in figure 6 is in P_1 as the initial marking of the net. $I(t_1) = \{P_1\}$, $I(t_2) = \{P_1\}$, $I(t_3) = \{P_1\}$, $I(t_4) = \{P_3\}$, $I(t_5) = \{P_3\}$, $I(t_6) = \{P_2, P_2\}$, $I(t_7) = \{P_2, P_2, P_2, P_2\}$, $I(t_8) = \{P_4, P_4\}$, $O(t_1) = \{P_1\}$, $O(t_2) = \{P_1\}$, $O(t_3) = \{P_2, P_2, P_3\}$, $O(t_4) = \{P_3\}$, $O(t_5) = \{P_2, P_2, \}$, $O(t_6) = \{P_2, P_2, \} O(t_7) = \{P_4, P_4\}$, $O(t_8) = \{P_5\}$. The weight of the arrow is shown by repeating the name of the place in the output / input function that many times. $TA = \{< A, P_1, A, 0 >\}$. The function ϕ shows the attributes of the tokens released after consuming the tokens from the input places. $F = P_5$.

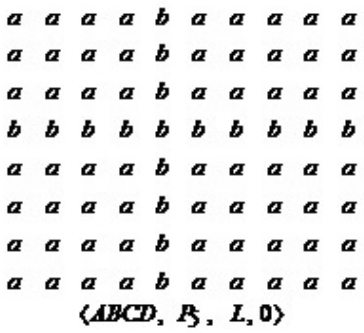

FIGURE 7. One of the arrays generated

$$
\begin{aligned}
\varphi_{t_1}(< A,\, P_1,\, A,\, 0 >) &= [< A,\, P_1,\, A \oplus a_m, :\, A,\, 0 >] \\
\varphi_{t_2}(< A,\, P_1,\, A,\, 0 >) &= [< A,\, P_1,\, A \otimes a^n, :\, A,\, 0 >] \\
\varphi_{t_3}(< A,\, P_1,\, A,\, 0 >) &= [< A,\, P_2,\, A,\, 20 >, < C,\, P_2,\, A : C,\, 20 >, < B,\, P_3,\, A : B,\, 0 >] \\
\varphi_{t_4}(< B,\, P_3,\, B,\, 0 >) &= [< B,\, P_3,\, B \otimes a^n, :\, B,\, 0 >] \\
\varphi_{t_5}(< B,\, P_3,\, B,\, 0 >) &= [< B,\, P_2,\, B,\, 0 >, < D,\, P_2,\, B :\, D,\, 0 >] \\
\varphi_{t_6}(TwoTokens) &= [< C,\, P_2,\, C \otimes a_m, :\, C,\, 0 >, < D,\, P_2,\, D \otimes a_m, :\, D,\, 0 >] \\
&\quad if\ id == \text{"}C\text{"}\ \&\&\ id == \text{"}D\text{"}
\end{aligned}
$$

$$
\begin{aligned}
\varphi_{t_7}(FourTokens) &= [< AB,\, P_4,\, A \otimes b^n \otimes B :\, M,\, 0 >, < CD,\, P_4,\, C \otimes b^n \otimes D :\, N,\, 0 >] \\
\varphi_{t_8}(TwoTokens) &= [< ABCD,\, P_4,\, M \oplus b_m \oplus N :\, L,\, 0 >]
\end{aligned}
$$

Firing t_1 adds a column of $a's$. Firing t_2 adds a row of $a's$. The transitions t_1 and t_2 can fire any number of times and they put the array back in P_1 without any delay. The result of firing these two transitions would give a rectangular array made of $a's$. The transition t_3 can fire without firing t_1 or t_2. The number written on any arc denotes the number of tokens consumed or generated. When t_3 fires, two copies of the rectangular array are put in the place P_2. One of it is stored as A and the other is stored as C. One more copy of the array is stored as B in P_3 . The token in place P_3 is immediately available but the two tokens A and C in P_2 are available only after a delay of 20 units of time. The tokens A and C wait for B and D to reach the place before t_6 can fire. When the transition t_5 fire, one token is consumed and two tokens are put in the place P_2. Two copies of the same array are put as B and D in P_2. The transition t_6 takes two tokens from P_2 and puts them back there after adding a column in both provided the tokens are having the id "C" and "D". If the tokens removed are having any other id, then t_6 does not fire. The transition t_7 consumes all four tokens A, B, C and D from P_2. It joins A and B row wise with a row of $b's$ in between them. It joins C and D row wise with a row of $b's$ in between them. These two new arrays are called M and N are put in P_4. The transition t_8 consumes the tokens M and N from P_4 joins them column wise with a column of $b's$ in between them. This new array which is put in P_5 is stored as L. The firing sequence is $t_1^2\, t_2\, t_3\, t_4\, t_5\, t_6\, t_7\, t_8$ generates the array in figure 7.

The language generated by the net in example 3 TCPNAL1 is the set of all arrays made of the letter "a" in which one column (neither the last nor the first) and one row (neither the last nor the first) are made of "b". This kind of language can be generated by a basic

Petri net, but it will be huge net with some inhibitor arcs used. Without using inhibitor arcs this control cannot be achieved. As an application, these types of generated arrays can be utilized as array type Password to improve the security of Password-Authentication process.

3. Conclusion

Petri net has been used to generate rectangular or hexagonal arrays. Huge Petri net may be required to generate certain array languages. Colored Petri net can also be used to generate arrays. In this paper time is taken as one of the four attributes of tokens. Function over the set of transitions is defined to describe the changes that are made in the attributes of the token, when firing a transition. All these additional characteristics introduced in the model are used effectively to generate arrays with minimum number of transitions and places. In the future, comparisons can be made between modules where time is an attribute of a transition and token. This Petri net and Colored Petri net models can be extended to do certain comparison in the generative capacity of various models.

References

[1] Peterson J. L., (1981), Petri Net Theory and the Modeling of Systems, Prentice Hall, Englewood Cliffs.
[2] James L. Peterson, (1980), A Note on Colored Petri Nets, Information Processing Letters, 11 (1), pp. 40-43.
[3] Jensen, Kurt, Rozenberg, Grzegorz (Eds.), (1991), High-level Petri Nets: Theory and Application, Springer Verelag.
[4] Jensen and Kurt, (1990), Coloured Petri Nets: A High Level Language for System Design and Analysis, in Advances in Petri Nets G. Rozenberg, ed., vol. 483 of Lecture Notes in Computer Science, Springer-Verlag, New York, pp. 342-416.
[5] Berthomieu, B. and Diaz, M., (1991), Modelling and verification of time dependent systems using time petri nets. IEEE Trans. on Software Engineering, 17 (3), pp. 259–273.
[6] Wang J., Deng Y and Xu G., (2000), Reachability Analysis of Real-time Systems Using Timed Petri Nets, IEEE Trans. on Syst. Man and Cybernetics, 30, (5), pp. 725-736.
[7] Jiacun W., (1998), Deterministic Timed Petri Nets, The International Series on Discrete Event Dynamic Systems, pp. 37-61.
[8] Zuberek, W. M., (1998), Timed Petri nets and performance evaluation of systems, Proceedings of The IEEE International Conference on Systems, Man, and Cybernetics, DOI: 10.1109/ICSMC.1998. pp. 725-422
[9] Wong C. Y., Tharam S. D., Forward, K. E., (1985), Timed Places Petri Nets with Stochastic Representation of Place Time, Proceeding of The International Workshop on Timed Petri nets, IEEE Computer Society, pp. 96-103.
[10] Van der Aalst W. M. P., (1992), Timed colored Petri nets and their application to logistics, Thesis, http://dx.doi.org/10.6100/IR381309.
[11] Van der Aalst W. M. P., (1993), Interval timed coloured petri nets and their analysis, International Conference on Applications and Theory of Petri Nets and Concurrency, Lecture Notes in Computer Science, 691, pp. 453-472.
[12] Holliday, M. A., Vernon, M. K., (1985), A Generalized Timed Petri Net Model for Performance Analysis, Proceedings of the International Workshop on Timed Petri Nets, pp. 181-190.
[13] Zuberek W. M., (1991), Timed Petri nets definitions, properties, and applications, 31, (4), pp. 627-644.
[14] Lalitha D., (2015), Rectangular array languages generated by a Colored Petri net, Proceedings of IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2015.
[15] Lalitha D., (2015). Rectangular array languages generated by a petri net, Advances in Intelligent Systems and Computing 332, Computational Vision and Robotics, pp. 17-27.
[16] Lalitha D., Rangarajan, K., Thomas D. G., (2012), Rectangular arrays and petri nets, International Workshop on Combinatorial Image Analysis, Lecture Notes in Computer Science, 7655, pp. 166-180.

[17] Lalitha D, and Rangarajan K., (2011), Petri Net Generating Hexagonal Arrays, International Workshop on Combinatorial Image Analysis, Lecture Notes in Computer Science, 6636, pp. 235-247.
[18] Lalitha D. and Rangarajan K., (2010), Column and row catenation petri net system,Bio-Inspired Computing: Theories and Applications, IEEE Fifth International Conference in 2010.
[19] Vaithyasubramanian S., Christy A. and Lalitha D., (2015), Two factor authentication for secured login using array password engender by Petri net, Procedia Computer Science, 48, pp. 313-318.
[20] Vaithyasubramanian S., Christy A. and Lalitha D., (2015), Generation of Array Passwords Using Petri Net for Effective Network and Information Security, Intelligent Computing, Communication and Devices, Advances in Intelligent Systems and Computing book series, 308, pp. 189-200.

Dr. S. Vaithyasubramanian started his teaching carrier in 2002. His interested areas of research are formal languages, Petri net, Mathematical Modelling, Information security and CAPTCHAs. He has published various articles in his fields of research in reputed journals.

Dr. D. Lalitha started her career in teaching as a Lecturer in Ethiraj College for women in 1986. Her field of research is Mathematical modelling with Petri nets, Formal Languages and Theory of Computation. She has published several research articles.

Mrs. M. I. Mary Metilda started her teaching carrier in 2005. Her interested areas of research are formal languages and Petri net. She is doing her research under the guidance of Dr. D. Lalitha.