The Application of Petri Nets to the Modelling and Analysis of a Producer-Consumer System

Ganiyu R. A
Department of Computer Science and Engineering,
Ladoke Akintola University of Technology, P.M.B 4000, Ogbomoso, Nigeria.

Omidiora E. O
Department of Computer Science and Engineering,
Ladoke Akintola University of Technology, P.M.B 4000, Ogbomoso, Nigeria.

Olabiyisi S. O
Department of Computer Science and Engineering,
Ladoke Akintola University of Technology, P.M.B 4000, Ogbomoso, Nigeria.

Okediran O. O
Department of Computer Science and Engineering,
Ladoke Akintola University of Technology, P.M.B 4000, Ogbomoso, Nigeria.

ABSTRACT

The present work deals with the application of Petri nets in modelling and analysis of a concurrent system – a producer-consumer system with a bounded buffer. In particular, a Petri net model was developed for the aforementioned system while the analysis of the developed model was carried out using reachability tree method in a bid to determine the behavioural properties of the modelled system. One of the major strengths of petri nets is their support for analysis of many properties associated with concurrent systems. Some of the properties that can be studied with a petri net model are those which depend on the initial marking. Such properties are referred to as behavoural properties. Thus, in this paper, through the analysis of the developed pertri net model, insights were gained into the behavioural properties of the producer-consumer system under consideration.

Keywords: Petri nets, model, producer-consumer system, analysis, reachability tree.

1. INTRODUCTION

Petri nets are a graphical tool for the formal description of the flow of activities in complex systems. With respect to other more popular techniques of graphical system representation (like block diagrams or logical trees), petri nets are particularly suited to represent in a natural way logical interactions among parts or activities in a system. Typical situations that can be modelled by petri nets are synchronization, sequentiality, concurrency and conflict (Bobbio, 1990). Petri nets provide a uniform environment for modeling, formal analysis, and design of discrete event systems (Zurwski and Zhoy, 1994). With Petri nets the main idea is to represent states of subsystems separately. Then, the distributed activities of a system can be represented very effectively (Hrúz and Zhou, 2007). Petri nets as graphical tools provide a powerful communication medium between the user, typically requirements engineer, and the customer. As a mathematical tool, a Petri net model can be described by a set of linear algebraic equations, or other mathematical models reflecting the behavior of the system (Zurwski and Zhoy, 1994; Hayder, 2008). Petri nets can be used by both practitioners and theoreticians. Thus, they provide a powerful medium of communication between them. Practitioners can learn from theoreticians how to make their models more methodical, and theoreticians can learn from practitioners how to make their models more realistic. However, Petri nets incorporate the fundamental concepts which can be as a basis for system designer and users who require new conceptual mechanisms and theories to deal with their systems. One of the major advantages of using Petri net models is that the same model is used for the analysis of behavioral properties and performance evaluation, as well as for systematic construction of discrete-event simulators and controllers (Zurwski and Zhoy, 1994; Lafta, 2005). The major weakness of Petri nets is the complexity problems. Petri net-based models tend to become too large for analyses (Murata, 1989).

Furthermore, the application of petri nets is through modeling. Petri net theory allows a system to be modelled by a petri nets. In many fields of study, a phenomenon is not studied directly but through a model of the phenomenon. A model is a representation, often in mathematical terms, of what are felt to be important features of the representation. By the manipulation of the representation, it is hoped that new knowledge about the modelled phenomenon can be obtained without the danger, cost or inconvenience of manipulating the real phenomenon itself. Analysis of the petri nets can, however, hopefully reveal important information about the structure and dynamic behaviour of the modelled system. This information can then be used to evaluate the modelled system and suggest improvements or changes.

2. METHODOLOGY

2.1 The Primitive Elements of a Petri Nets

For definitions and notation we refer in general to (Peterson, 1981). A Marked PN is a quintuple (*P, T, I, O, M*), where:

- $P=\{p_1, p_2, \ldots, p_{np}\}$ is the set of n_p places (drawn as circles in the graphical representation);
- $T=\{t_1, t_2, \ldots, t_{nt}\}$ is the set of n_t transitions (drawn as bars);
- *I* is the transition input relation and is represented by means of arcs directed from places to transitions;
- *O* is the transition output relation and is represented by means of arcs directed from transitions to places;
- $M=\{m_1, m_2, \ldots, m_{np}\}$ is the marking. The generic entry m_i is the number of tokens (drawn as black dots) in place p_i in marking *M*.

The graphical structure of a PN is a bipartite directed graph: the nodes belong to two different classes (places and transitions) and the edges (arcs) are allowed to connect only nodes of different classes (multiple arcs are possible in the definition of the *I* and *O* relations (Peterson, 1981). The behaviour of many systems can be described in terms of system states and their changes. In order to simulate the dynamic behaviour of a system, a state or marking in a Petri nets is changed according to the following transition (firing) rule:

- A transition t is said to be enabled if each input place *p* of *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 fire (depending on whether or not the event actually takes place).
- 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* (Murata, 1989).

The above transition rule is illustrated in Figures 2.1 and 2.2 respectively. The petri nets shown in Figure 2.1 contains the following:

- Two input places *p1* and *p2*
- One output place *p3*
- Transition *t*

- Arc from *p1* to *t*
- Arc from *p2* to *t*
- Arc from *t* to *p3*
- Weight of the arc from *p1* to *t* which is equal to 2 (i.e. *w(p1, t)* = 2)
- Weight of the arc from *p2* to *t* which is equal to 1 (i.e. *w(p2, t)* = 1)
- Weight of the arc from *t* to *p3* which is equal to 2 (i.e. *w(t, p3)* = 2)
- Two tokens in input places *p1* (represented by the two black dots)
- Two tokens in input places *p2* (represented by the two black dots)

With the presence of two tokens in each input place, the transition *t* is enabled. After firing *t*, the marking (2, 2, 0), which is the initial marking obtained from number of tokens residing in places *p1*, *p2* and *p3*, will change to the marking (0, 1, 2) shown in Figure 2.2 where the transition *t* is no longer enabled.

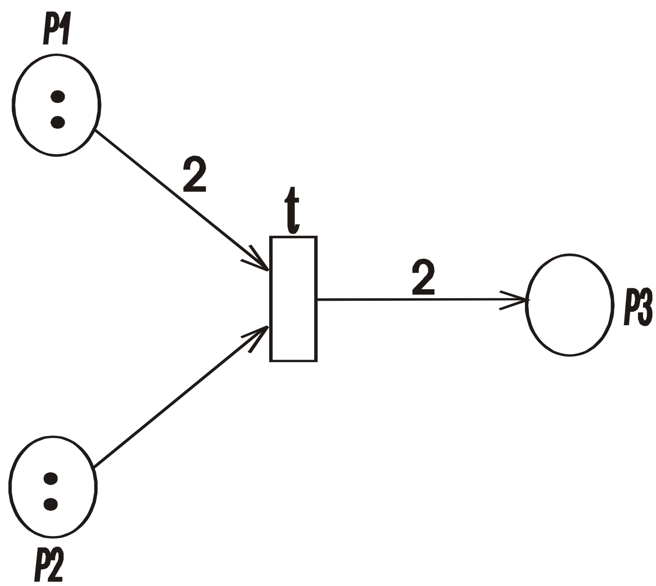

Figure 2.1: The marking before firing the enabled transition *t*.

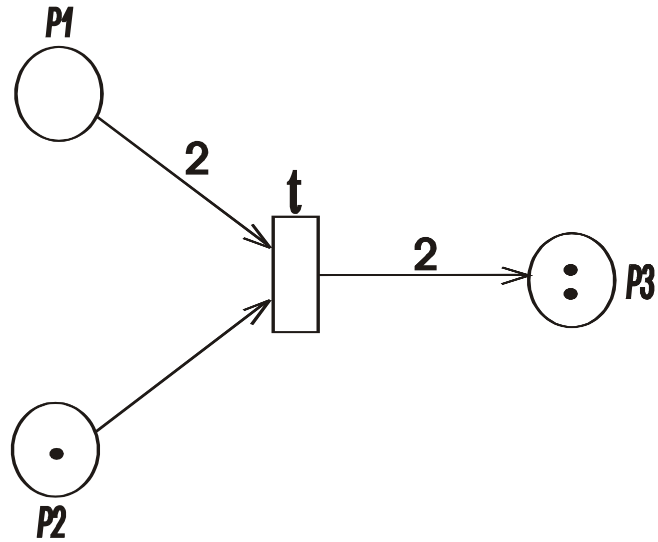

Figure 2.2: The marking after firing *t*, where *t* is disabled.

2.2 Properties of Petri Nets

A major strength of Petri nets is their support for analysis of many properties and problems associated with concurrent systems (Murata, 1989). These properties when interpreted in the context of the model system, allow the system designer to identify the presence or absence of the application domain specific functional properties of the system under design. Two types of properties can be distinguished: behavioral and structural properties (Zurwski and Zhoy, 1994).

2.2.1 Behavioral Properties.

The behavioral properties are those which depend on the initial state, or marking of a Petri net (Zurwski and Zhoy, 1994). These include the following:

2.2.1.1 Reachability

Reachability is a fundamental basis for studying the dynamic properties of any system. An important issue in designing distributed system is whether a system can reach a specific state, or exhibit a particular functional behavior (Murata, 1989). During execution different markings can be reached in a Petri net. The markings are either desirable or undesirable from the viewpoint of the operation of the modeled system. The reachability problem addresses the equation whether it is possible to reach or avoid a given marking starting from a given initial state (Cassandras and Lafortune, 2008).

2.2.1.2 Liveness

The liveness property is one of the most important properties of the Petri net for most applications. The liveness property encapsulates the concept of a system which will be able to run continuously, i.e. a system which does not deadlock, which is an important property when modelling operating systems, communication protocols, computer programs, and just about any safety critical system. Using the most general definition of liveness, a Petri net is considered live if, for all possible makings, there is always a transition enabled. There have been, however, many other concepts developed in the area of liveness, some of which define different levels of liveness for individual transitions.

2.2.1.3 Safeness

The property of safeness can be determined for both individual places and for the entire net. A place is said to be safe, if for all possible makings, the number of tokens in that place never exceeds one. The petri net is declared safe if all the places in the net are safe.

2.2.1.4 Boundedness

The property of boundedness can also be determined for individual places and for the entire Petri net. The boundedness property is a more general form of the saveness property. A place is said to be k-bounded if, for all possible markings, the number of tokens in that place never exceeds k. A Petri net is k-bounded if, for all possible markings, the number of tokens in any individual place in the net never exceeds k. Both the safe and bound properties for petri nets are important in the field of engineering because they help determine what size buffers, counter, etc are needed in order to implement the design. For instance, if all of the places are safe, it would be possible to implement these conditions with a Boolean variable. If a place is representative of a buffer, and is 20-bounded, then the designer knows to use a buffer of size 20.

2.2.1.5 Reversibility and Home state

An important issue in the operation of real systems, such as manufacturing systems, process control systems, etc., is the ability of these systems for an error recovery (Zurwski and Zhoy, 1994). The reversibility property says that for every reachable marking there exists at least one is firing sequence beginning at it and going back to the initial marking (Hrúz and Zhou, 2007). In many applications, it is not necessary to get back to the initial state as long as one can get back to some (home) state (Murata, 1989).

2.2.1.6 Coverability

A marking M in a Petri net (N, M_0) is said to be coverable if there exists a marking M' in reachability set of M_0 such that $M'(p) \geq M(p)$ for each p in the net (Murata, 1989; Hangos, Lakner and Gerzson, 2004).

2.2.1.7 Persistence

A Petri net is said to be persistent if for any two enabled transitions, the firing of one cannot disable the other (Cassandras and Lafortune, 2008).

2.2.1.8 Fairness

A property closely related to the persistence is fairness. A Petri net is said to be bounded fair if every pair of its transitions is bounded fair. A transition pair is bounded fair if every transition of the pair can fire K-times at maximum before the other transition in the pair fires (Hrúz and Zhou, 2007).

2.2.2 Structural Properties

Structural properties are those which depend on the topological structures of Petri nets. They are independent of the initial marking M_0 in the sense that these properties hold for any initial marking or are considered with the existence of certain firing sequences from some initial marking. Thus, these properties can often be characterized in terms of the incidence matrix A and its associated homogenous equations or inequalities (Murata, 1989; Zurwski and Zhoy, 1994).

2.2.2.1 Structural liveness

Petri net with initial marking is said to be live if there always exists some sample path such that any transition can eventually fire from any state reached from initial marking (Cassandras and Lafortune, 2008).

2.2.2.2 Controllability

A Petri net is said to be completely controllable if any marking is reachable from any other marking (Murata, 1989; Lafta, 2005).

2.2.2.3 Structural Boundedness

A Petri net is structurally bounded if it is bounded given any initial marking M_0 (Hrúz and Zhou, 2007). It is structurally bounded if and only if there exist an *m-vector y* of positive integers such that $Ay \leq 0$. The number of tokens in each place *P*, is bounded by: $M(p) \leq M_0^T y / y(p)$, where *y(p)* is the p_{th} entry of *y* (Murata, 1989; Gomes, Costa, Barros, Pais, Rodrigues and Ferreira, 2007).

2.2.2.4 Conservativeness

A Petri net is (partially) conservative if and only if there exists an *m-vector* of positive (nonnegative) integers such that $Ay=0$, $y \neq 0$ (Gomes, Costa, Barros, Pais, Rodrigues and Ferreira, 2007).

2.2.2.5 Repetitiveness:

A Petri net is said to be (Partially) repetitive if there exists an *n-vector x* of positive integers such that $A^T x \geq 0$, $x \neq 0$ (Murata, 1989).

2.2.2.6 Consistency:

A Petri net is said to be (partially) consisted if and only if there exists an *n-vector x* of positive integers such that $A^T x = 0$, $x \neq 0$ (Murata, 1989).

2.3 Petri Nets and the Modelling of a Producer-Consumer System

Petri nets used for modelling real systems are sometimes referred to as Condition/Events nets. Places identify the conditions of the parts of the system (working, idle, queueing, failed, etc), and transitions describe the passage from one condition to another (end of a task, failure, repair, etc). An event occurs (a transition fires) when all the conditions are satisfied (input places are marked) and give concession to the event. Occurrence of the event modifies in whole or in part the status of the conditions (marking). The number of tokens in a place can be used to identify the number of resources lying in the condition denoted by that place (Bobbio, 1990). In modelling, using the concept of conditions and events, places represent conditions, and transitions represent events. A transition (event) has a certain number of input and output places representing pre-conditions and post-conditions of the events, respectively. The presence of a token in a place is interpreted as holding the truth of the condition associated with the place. In another interpretation, k tokens are put in a place to indicate that k data items or resources are available (Murata, 1989).

In a producer-consumer system, a producer produces items that are put into a buffer from which they can be removed and consumed by a consumer. Figure 2.3 shows the developed petri net model of a producer-consumer system with a bounded buffer. On the left is a producing entity, and on the right is the consumer entity. In the center is a buffer which allows up to twenty items to be produced at a time. Once the producer produces twenty items, the buffer is full and the producer will not be able to produce any more items until the consumer consumes at least one item. The petri net model has a set of six transitions labeled T1 through T6. All the arcs in this model are enabled by a single token. The initial marking of this petri net is (1, 0, 0, 0, 20, 1, 0, 0) which is determined by noting the number of tokens in each place from P1 to P8 respectively. Notice that the number of tokens in each place is noted under the place with a text label "TK=1" to represent one token. If the place contains no tokens, the label is omitted.

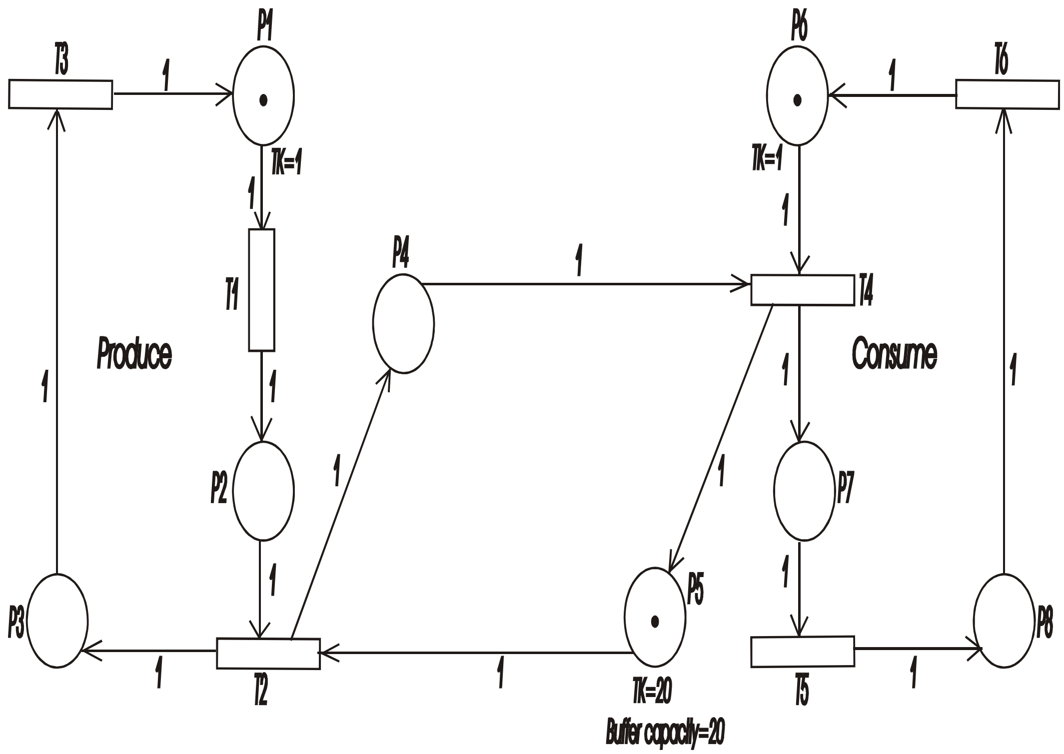

Figure 2.3: The developed petri net model of a producer-consumer system with a bounded buffer.

2.4 Analyzing the developed petri net model of a producer-consumer system

Once a petri net has been developed, it is pertinent to be able to analyze the net to determine what sort of properties it has in order to determine if the design will be feasible for a particular application. Nevertheless, methods of analysis for Petri nets can be classified into three groups: (1) the coverabihty or the reachability tree method, (2) the matrix-equation approach, and (3) reduction or decomposition techniques (Murata, 1989). Even though the reachability and coverabihty tree remain as effective methods for computing optimal legal firing sequences (LFSs) of PNs, it cannot be used for unbounded PNs due to the presence of the pseudo-infinity symbol ω (Desrochers and Al-Jaar, 1995). The use of the reachability tree is too limited due to the state-space-explosion for large problems. Reduction or decomposition techniques are too theoretical and lack practical applicability. This limits the use of decomposition techniques. The matrix-equation approach comes out to be the most useful and reliable technique due to its mathematical rigor, rigidity and wide applicability. This is a quite different technique for analyzing PNs based on matrix linear algebra (Murata, 1989). The advantage of the matrix-equation techniques over the reachability-tree or the coverability-tree technique is the existence of simple linear-algebraic equations that aid in determining PN properties (Ahmed, 2001).

While several approaches to the analysis of Petri nets have been considered, almost all work in this area eventually uses one basic technique (Peterson, 1977). This technique involves finding a finite representation for the reachability set of a Petri net, in recognition of the fact that many of the properties of a Petri net are based on properties of its reachability set. The representation used is known as the reachability tree. It consists of a tree whose nodes represent markings of the Petri net and whose arcs represent the possible changes in state resulting from the firing of transitions (Karp and Miller, 1969; Keller, 1972).

Notice, however, that the reachability set of a marked Petri net is often infinite. Thus, to form a finite representation of an infinite set we must map many markings into the same node of the tree. This many-to-one mapping is accomplished by collapsing a set of states into a node by ignoring the number of tokens in a place of the net when this number becomes "too large." This is represented by using a special symbol, ω, for the number of tokens in this place. The symbol ω represents a value which can be arbitrarily large. Each node in the reachability tree is labeled with a marking; arcs are labeled with transitions. The initial node (root of the reachability tree) is labeled with the initial marking. Given a node x in the tree, additional nodes are added to the tree for all markings that are directly reachable from the marking of the node x. For each transition t_j which is enabled in the marking for node x, a new node with marking $\delta(x, t_j)$ is created, and an arc labeled t, is directed from the node x to this new node. This process is repeated for all new nodes.

Continuing this process will obviously create the entire state-space. A path from the initial marking (root) to a node in the tree corresponds to an execution sequence. Since the state-space may be infinite, two special steps are taken to define a finite reachability tree. First, if a new marking is generated which is equal to an existing marking on the path from the root node to the new marking, the new (duplicate) marking becomes a terminal node. Since the new marking is equal to the previous marking, all markings reachable from it have already been added to the reachability tree by the earlier identical marking. Secondly, if any new marking x is generated which is greater than a marking y on the path from the root node to the marking x, then those components of marking x which are strictly greater than the corresponding components of marking y are replaced by the symbol ω. Since marking x is greater than marking y, any sequence of transition firings which is possible from marking y is also possible from marking x. In particular, the sequence that transformed marking y into marking x can be repeated indefinitely, each time increasing the number of tokens in those places which have a ω. Thus the number of tokens in these places can be made arbitrarily large.

By considering the developed petri net model of a producer-consumer system shown in Figure 2.3, the analysis step usually consists of forming a reachability tree. A portion of the reachability for the petri net developed model (Figure 2.3) is as depicted in Figure 2.4. The reachability tree begins with the initial marking (1, 0, 0, 0, 20, 1, 0, 0). From this marking, a new marking is created for each enabled transition from that marking. In this case, there is only one enable transition T1, which results in a marking of (0, 1, 0, 0, 20, 1, 0, 0). Subsequently, a new marking is created for each enabled transition from the marking (0, 1, 0, 0, 20, 1, 0, 0), and the process repeats. Once a marking is found to be already present in the tree, it is marked with an asterisk and that marking need not be added again. The completed reachability tree shows all the markings that can be obtained from the initial marking given every possible sequence of transition firings.

From the partial reachability tree shown in Figure 2.4, it is clearly appeared that place P5 is not safe since, for all possible markings, the number of tokens contained within that place exceeds one. Hence, the petri net cannot be declared safe. This petri net is 20-bounded since, for all possible markings, the number of tokens in any individual place in the net never exceeds 20. Also, this petri net is considered live since, for all possible markings, there is always a transition enabled. Another important property for petri net analysis is conservativeness. This petri net is strictly conservative since, for all markings the total sum of all tokens always equal to twenty two.

(1, 0, 0, 0, 20, 1, 0, 0)	T1 →	(0, 1, 0, 0, 20, 1, 0, 0)
(0, 1, 0, 0, 20, 1, 0, 0)	T2 →	(0, 0, 1, 1, 19, 1, 0, 0)
(0, 0, 1, 1, 19, 1, 0, 0)	T3 →	(1, 0, 0, 1, 19, 1, 0, 0)
	T4 →	(0, 0, 1, 0, 20, 0, 1, 0)
(1, 0, 0, 1, 19, 1, 0, 0)	T1 →	(0, 1, 0, 1, 19, 1, 0, 0)
	T4 →	(1, 0, 0, 0, 20, 0, 1, 0)
(0, 0, 1, 0, 20, 0, 1, 0)	T3 →	(1, 0, 0, 0, 20, 0, 1, 0)
	T5 →	(0, 0, 1, 0, 20, 0, 0, 1)
(0, 1, 0, 1, 19, 1, 0, 0)	T2 →	(0, 0, 1, 2, 18, 1, 0, 0)
	T4 →	(0, 1, 0, 0, 20, 0, 1, 0)
(1, 0, 0, 0, 20, 0, 1, 0)	T1 →	(0, 1, 0, 0, 20, 0, 1, 0)
	T5 →	(1, 0, 0, 0, 20, 0, 0, 1)
(0, 0, 1, 0, 20, 0, 0, 1)	T3 →	(1, 0, 0, 0, 20, 0, 0, 1)
	T6 →	(0, 0, 1, 0, 20, 1, 0, 0)
(0, 0, 1, 2, 18, 1, 0, 0)	T3 →	(1, 0, 0, 2, 18, 1, 0, 0)
	T4 →	(0, 0, 1, 1, 19, 0, 1, 0)
(0, 1, 0, 0, 20, 0, 1, 0)	T2 →	(0, 0, 1, 1, 19, 0, 1, 0)
	T5 →	(0, 1, 0, 0, 20, 0, 0, 1)
(1, 0, 0, 0, 20, 0, 0, 1)	T1 →	(0, 1, 0, 0, 20, 0, 0, 1)
	T6 →	(1, 0, 0, 0, 20, 1, 0, 0)
(0, 0, 1, 0, 20, 1, 0, 0)	T3 →	(1, 0, 0, 0, 20, 1, 0, 0)
(1, 0, 0, 2, 18, 1, 0, 0)	T1 →	(0, 1, 0, 2, 18, 1, 0, 0)
	T4 →	(1, 0, 0, 1, 19, 0, 1, 0)
(0, 0, 1, 1, 19, 0, 1, 0)	T3 →	(1, 0, 0, 1, 19, 0, 1, 0)
	T5 →	(0, 0, 1, 1, 19, 0, 0, 1)
(0, 1, 0, 0, 20, 0, 0, 1)	T2 →	(0, 0, 1, 1, 19, 0, 0, 1)
	T6 →	(0, 1, 0, 0, 20, 1, 0, 0)

Figure 2.4: Partial reachability tree for the developed Petri net model depicted in Figure 2.3

3. DISCUSSION OF RESULTS

In this paper, we have been able to develop a Petri net model of the producer-consumer system with a bounded buffer. The analysis of the developed Petri net model was carried out using reachability tree method in a bid to gain insights into the behavioural properties of the modelled producer-consumer system. One of the major strengths of petri nets is their support for analysis of many properties associated with concurrent systems. Some of the properties that can be studied with a petri net model are those which depend on the initial marking. Such properties are referred to as behavoural properties. From the partial reachability tree created for the developed petri net model of a producer-consumer system under consideration, it is clearly appeared that place P5 is not safe since, for all possible markings, the number of tokens contained within that place exceeds one. Hence, the petri net cannot be declared safe. Furthermore, the petri net is 20-bounded since, for all possible markings, the number of tokens in any individual place in the net never exceeds 20. Also, this petri net is considered live since, for all possible markings, there is always a transition enabled. Another important property for petri net analysis is conservativeness. In the same vein, the petri net is strictly conservative since, for all markings the total sum of all tokens always equal to twenty two.

4. CONCLUSION

The ability to analyze petri net is generally considered to be the most important activity. Once a petri net has been developed, it is pertinent to analyze the net to determine what sort of properties it has in order to determine if the design will be feasible for a particular application. Through this medium, in a bid to gain insights into the behavioural properties of the modelled producer-consumer system under consideration, the analysis of the developed Petri net model was explored using reachability tree analysis method. Having constructed the reachability tree, insights were gained into the behaviours and important properties (i.e. safeness, boundedness, liveness and conservativeness properties) of the modelled producer-consumer system. Albeit, future research may be geared towards analyzing the developed Petri net model of a producer-consumer system under consideration, which is characterized by a bounded buffer, using the matrix-equation approach and reduction or decomposition techniques.

5. REFERENCES

Ahmed, T. (2001): “Optimization Algorithms Applied to Large Petri Nets”, PhD dissertation in Computer Science, Graduate Faculty of Texas Tech University.

Bobbio, A. (1990): “System Modelling with Petri Nets”, A.G. Colombo and A. Saiz de Bustamante (eds.), System Reliability Assessment, Kluwer p.c., pp 102-143.

Cassandras, C. G. and Lafortune, S. (2008): "Introduction to Discrete Event Systems", Springer Science+Business Media, LLC.

Desrochers, A. A. and Al-Jaar, R.Y. (1995): “Applications of Petri Nets in Manufacturing Systems: Modeling, Control, and Performance Analysis”, IEEE Press, IEEE Control Systems, Piscataway, New Jersey.

Gomes, L., Costa, A., Barros, J. P., Pais, R., Rodrigues, T. and Ferreira, R. (2007): "Petri Net based Building Automation and Monitoring System", Copyright in IEEE, pp. 57-62.

Hangos, K. M., Lakner, R. and Gerzson, M. (2004): "Intelligent Control Systems", Kluwer Academic Publishers.

Hayder, H. S. (2008): “The Implementation of Petri Net to the Modeling, Analysis, and Control”, M.Sc thesis in Control and Computer Engineering, College of Engineering, University of Baghdad.

Hrúz, B. and Zhou, M. C. (2007): "Modeling and Control of Discrete-event Dynamic Systems with Petri Nets and Other Tool", Springer-Verlag London Limited.

Karp, R. M. and Miller, R. E. (1969): “Parallel program schemata”, J. Computer and Systems Science, Vol. 3, No. 4, pp. 167-195.

Keller, R. M. (1972): “Vector replacement systems: a formalism for modelling asynchronous systems”, Tech. Rep. 117, Computer Sci. Laboratory, Princeton Univ., Princeton, N. J.

Lafta, H. A. (2005): "Supect identification using Fuzzy Petri nets", PhD thesis, University of Technology, Computer Science.

Murata, T. (1989): "Petri nets: properties, analysis and application", proceedings of the IEEE, Vol. 77, No. 4, pp. 541-580.

Peterson, J. L. (1977): “Petri nets”, Computing Surveys, Vol. 9, pp. 223‐ 252.

Peterson, J.L. (1981): “Petri Net Theory and the Modeling of Systems”, Prentice Hall, Englewood Cliffs, New Jersey.

Zurwski, R. and Zhoy, M. (1994): "Petri nets and Industrial Applications: A Tutorial", IEEE Transaction on Industrial Electronics, Vol. 41, No. 6, pp. 567-583.

ABOUT THE AUTHORS

Ganiyu R. A. is a lecturer in the Department of Computer Science and Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria. He graduated with B.Tech. Computer Engineering and M. Tech. Computer Science from Ladoke Akintola University of Technology, Ogbomoso, Nigeria, in 2002 and 2008 respectively. He has almost finished his Ph.D Computer Science in the same Institution. He has published in reputable journals. His research interests include: Dynamic Programming and their Applications; Theoretical Computer Science; Modelling and Simulation of Concurrent Systems Using Petri Nets (Low level and High level). He belongs to the following professional bodies: Full member, Computer Professionals (Registration) Council of Nigeria (MCPN); Registered Engineer, Council for the Regulation of Engineering in Nigeria (COREN).

Omidiora E. O. is currently a lecturer in the Department of Computer Science and Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria. He graduated with B.Sc. Computer Engineering (1991) from Obafemi Awolowo University, Ile-Ife, Nigeria. He bagged M.Sc. Computer Science from University of Lagos, Nigeria (1998) and Ph.D Computer Science from Ladoke Akintola University of Technology (2006). He has published in reputable journals and learned conferences. His research interests include: The study of Biometric Systems, Computational Complexity measures and Soft Computing. He belongs to the following professional bodies: Full Member, Computer Professionals (Registration) Council of Nigeria; Corporate Member, Nigeria Society of Engineers; Register Engineer, COREN etc.

Olabiyisi S. O. received B. Tech., M. Tech and Ph.D degrees in Mathematics from Ladoke Akintola University of Technology, Ogbomoso, Nigeria, in 1999, 2002 and 2006 respectively. He also received M.Sc. degree in Computer Science from University of Ibadan, Ibadan, Nigeria in 2003. He is a lecturer in the Department of Computer Science and Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria. He has published in reputable journals and learned conferences. Dr Olabiyisi is a member of Computer Professional (Registration) Council of Nigeria (CPN). His research interests are in Computational Mathematics, Computational Complexity, Theoretical Computer Science, Simulation and Performance Evaluation.

Okediran O. O. is a lecturer in the Department of Computer Science and Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria. He graduated with B.Tech. Computer Engineering and M. Tech. Computer Science from Ladoke Akintola University of Technology, Ogbomoso, Nigeria, in 2002 and 2008 respectively. He has almost finished his Ph.D Computer Science in the same Institution. He has published in reputable journals. His research interests include: Computational optimization, e-commerce, biometrics-based algorithms and its applications to e-voting systems. He belongs to the following professional bodies: Full member, Computer Professionals (Registration) Council of Nigeria (MCPN); Registered Engineer, Council for the Regulation of Engineering in Nigeria (COREN).