PIPE PLUGIN FOR HYBRID REWRITING PETRI NETS VISUAL SIMULATION OF WIRELESS SENSOR NETWORKS

Andrei FURTUNA*

Technical University of Moldova, Faculty of Computers, Informatics and Microelectronics, Chișinău, Republica Moldova

*Corresponding author: Andrei Furtuna, furtuna_andrei@outlook.com

Abstract. *Wireless Sensor networks (WSNs) have become one of the most interesting areas of research in the past few years. This paper presents an overview of proposed plugin for adding feature of modeling hybrid rewritable petri nets for Platform-Independent Petri Net Editor (PIPE), an open-source tool that supports the design and analysis of Generalized Stochastic Petri Net (GSPN) models. Recent advances in wireless and electronic technologies have enabled a wide range of applications of WSNs in military, traffic surveillance, target tracking, environment monitoring and healthcare monitoring. Petri nets are a widely used formalism for the analysis of concurrent systems and as such there are a lot of existing tools which allow users to edit, animate and analyze a range of Petri net classes. PIPE 's extensible design enables developers to add functionality via pluggable analysis modules. It also acts as a front-end for a parallel and distributed performance evaluation environment.*

Key words: *Wireless sensor networks,* rewriting *hybrid Petri Nets, computer system*

1. Introduction

Last decades there is a big trend of smart gadgets, which in main part are based on sensors. WSNs have become the one of the most interesting areas of research in the past few years. WSNs are usually composed of small, low cost devices that communicate wirelessly and have the capabilities of processing, sensing and storing. A WSN generally consists of a base station (also called as gateway sometimes) that can communicate with a number of wireless sensors via a radio link. Wireless sensor nodes collect the data, compress it, and transmit it to the gateway directly or indirectly with the help of other nodes. The transmitted data is then presented to the system by the gateway connection. This paper discusses the recent advances in WSNs that enable a wide range of applications and future development in applications like underwater acoustic sensor systems; sensing based cyber physical systems, time critical applications, cognitive sensing and spectrum management, and security and privacy management [1,2].

Petri nets (PN) are a well-accepted formalism for modelling concurrent and distributed systems in various application areas: workflow management, embedded systems, production systems, sensor networks and traffic control are but a few examples. The main advantages of PNs are their graphical notation, their simple semantics, and the rich theory for analyzing their behaviour. In the current article we propose modeling and simulation of a rewritable network, this is very similar to real-time wireless sensor networks. For instance, imagine if a node has issues, then we can easily reconfigure another one to do same functionality as the broken sensor. The idea of rewriting rules is also described in [3] where GSPN can dynamically modify their own structures by rewriting rules transitions some of their components thus supporting structural dynamic changes within modeled systems.

Performance verification and evaluation are some of techniques widely used to design and analyze different distributed systems and hybrid applications with discrete-continuous processes. Using this techniques design, gaps can be identified at the beginning stage [4-8]. Thus these problems can be eliminated earlier, and troubleshooting, maintenance and maintenance costs can be significantly reduced.

In spite of their graphical nature, getting an understanding of a complex system just from studying the Petri net model itself is quite hard – if not impossible. In particular, this applies to experts from some application area who, typically, are not experts in PNs [9].

Platform-Independent Petri Net Editor (PIPE) is a Java-based tool for the construction and analysis of Generalised Stochastic PN (GSPN) models. PIPE began life in 2002/3 as a postgraduate team programming project in the Department of Computing at Imperial College London and has been steadily improved through a number of successive versions, implemented by students at Imperial College London and with input from industry (particularly Intelligent Automation, Inc.) In addition, a branched version with significant improvements to different aspects of functionality (e.g. the addition of inhibitor arcs and fixed-capacity places, and an experimental framework) has also been implemented at the Universitat de les Illes Balears [10].

In this paper we propose to add the feature for modeling and simulating dynamic rewriting hybrid PNs (RHPN) in the above-mentioned software. This will facilitate the analysis, visualization and simulation of further work in domain of Wireless Sensor Networks, which is one of the top trends from last decades.

In Section 2 we briefly introduce the definition of RHPNs. We then proceed, in Section 3, to present the PIPE tool and the way of adding new functionality. Last, the 4th section is reserved for conclusion.

2. Descriptive dynamic rewriting hybrid Petri Nets

The theory of PNs has its origin in C.A. Petri's dissertation "Communication with Automata", submitted in 1962. PNs are used as describing formalism in a wide range of application fields. They offer formal graphical description possibilities for modeling systems consisting of concurrent processes. PNs extend the automata theory by aspects like concurrency and synchronization.

Due to last trends the capability of reconfiguration separate devices/nodes is giving a wide control on PN topology. Previous works in this direction were done mainly in fields of process control, automation and distributed systems [3, 4]. We decided to apply these techniques to WSN, which suits very good for real-time network control. In the following we demonstrate the possibilities of using RHPNs to model wireless sensor networks infrastructure, which is an embedded hybrid system. The used PN class of Hybrid Dynamic Nets (HDN) and its object oriented extension is described in [3, 8, 11, 12]. This class is derived from the above-mentioned approach of David and Alla and defines the firing speed as function of the marking from the continuous net places.

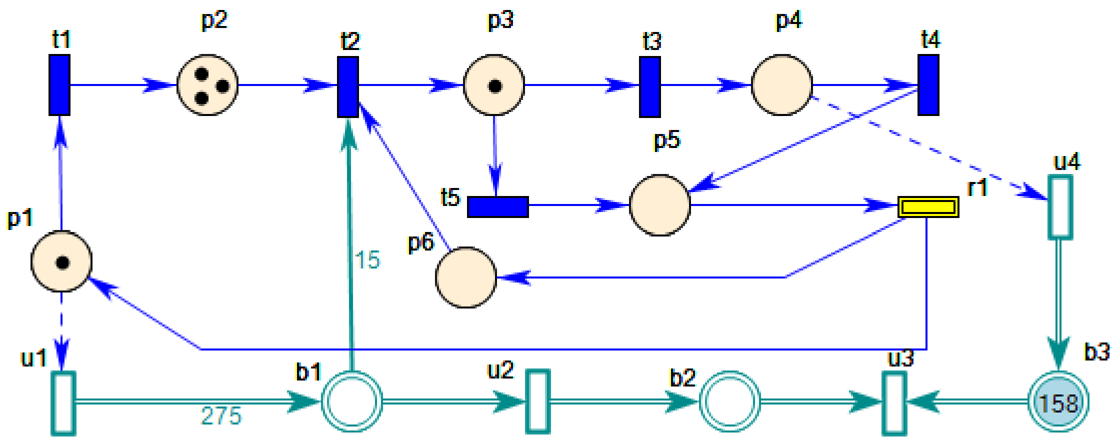

Figure 1: Example of Rewriting Hybrid petri net

Continuous PNs are particularly suitable for modeling flows: liquid flow, data flows (suitable for WSN), continuous production of a machine. However, a flow may be suddenly interrupted: closing a valve or machine breakdown for example. This is equivalent to suddenly having another continuous PN. This situation can be modeled by a RHPN containing continuous places and transitions (**C-places and C-transitions**) and discrete places and transitions (**D-places and D-transitions**). In addition, in a RHPN, a discrete marking may be converted into a continuous marking and vice-versa. The marking of a C-place is represented by a real number, whose unit is called a mark, and the marking of a D-place is represented by dots, called *tokens* (or *marks* when a common word is useful). In Figure 1 is represented an example of RHPN, where transition **t1** is enabled only if there is at least one token in **p1**.

Formal Definition

A marked **autonomous rewriting hybrid Petri net** is a tuple $RN = \langle P, T, Tr, Pre, Post, m_0, h\rangle$ fulfilling the following conditions:

$P = \{P_1, P_2, \dots, P_n\}$ is a finite, not empty, set of places;
$T = \{T_1, T_2, \dots, T_m\}$ is a finite, not empty, set of transitions;
$Tr = \{Tr_1, Tr_2, \dots, Tr_m\}$ is a finite, not empty, set of rewriting transitions;
$P \cap T = \emptyset$, $T \cap Tr = \emptyset$ and $P \cap Tr = \emptyset$, i.e. the sets *P, T* and Tr are disjointed;
$h: P \cup T \cup Tr \to \{D, C\}$, called "hybrid function", indicates for every node whether it is a discrete node (sets P^D and T^D) or a continuous one (sets P^C and T^C);
$Pre: P \times T \times Tr \to Q_+$ or N is the input incidence application;
$Post: P \times T \times Tr \to Q_+$ or N is the output incidence application;
$m_0: P \to R_+$ or N is the initial marking

In the definitions of $Pre, Post,$ and m_0, N corresponds to the case where $P_i \in P^D$, and Q_+ or R_+ corresponds to $P_i \in P^C$. Pre and $Post$ functions must meet the following criterion: if P_i and T_j are such that $P_i \in P^D$ and $T_i \in T^D$, then $Pre(P_i, T_j) = Post(P_i, T_j)$ must be verified.

As for a discrete or a continuous PN, the structure (implying the incidence matrix) is defined by the quadruple $\langle P, T, Pre, Post\rangle$. An *unmarked hybrid PN* is a 6-tuple $Q = \langle P, T, Tr, Pre, Post, h\rangle$, i.e., in addition to the structure, the nature of each node (discrete or continuous) is specified in *Q*.

The last condition states that an arc must join a C-transition to a D-place as soon as a reciprocal arc exists. This ensures marking of D-places to be an integer whatever evolution occurs.

Definition 2.

A **discrete transition** in a hybrid PN is **enabled** if each place P_i in $^*T_j$ meets the condition (as for a discrete PN):

$$m(P_i) = Pre(P_i, T_j).$$

Definition 3.

A **continuous transition** in a hybrid PN is **enabled** if each place P_i in $^*T_j$ meets the condition (as for a discrete PN):

$$\begin{cases} m(P_i) \geq Pre(P_i, T_j), & \text{if } P_i \text{ is a D-place} \\ m(P_i) > 0, & \text{if } P_i \text{ is a C-place} \end{cases}$$

3. PIPE modelling software

Models are drawn on a canvas using components from a drawing toolbar including places, transitions, arcs and tokens Figure 3.1. Nets of arbitrary complexity can be drawn and annotated with additional user information. Besides basic model design functionality, the designer interface provides features such as zoom, export, tabbed editing and animation. The animation mode is particularly useful for aiding users in the intuitive verification of the behaviour of their models. PIPE uses the Petri Net Markup Language (PNML) [15] as its file format, which permits interoperability with other Petri net tools including P3 [16], WoPeD [17] and Woflan [18] and the models are saved into XML files.

The project is maintained in programming language Java 8, and has a wide community of users. The source code of the project is available on GitHub versioning server. For project building automation is used Maven which permits to add easily a new module (in our case – the proposed plugin).

Conclusion

The feature of modeling RHPNs, brings PIPE simulation and evaluation tool to a larger space of problems. Rewrite option permits to have a dynamic topology of wireless sensor networks, which permits to reconfigure individual nodes. One of problems is WSN which we propose to evaluate. We are currently implementing and testing this plugin. Future work on this subject can be, graphical representation of analysis, in form of graphs.

References

1. Gupta, Dr. Deepti. "Wireless Sensor Networks 'Future trends and Latest Research Challenges'." ISSN: 2278-8735, pp. 41-46, (2015).
2. Yang, Yuanyuan & Wang, Cong & Li, Ji. Wireless Rechargeable Sensor Networks - Current Status and Future Trends. Journal of Communications. 10. 696-706. 10.12720/jcm.10.9.696-706, (2015).
3. Guțuleac E., Mocanu M., Descriptive dynamic rewriting GSPN-based performance modeling of computer systems. Proceding of the 15th International Conference on Control Systems and Computer Science,CSCS15, pp.656-661, (2005).
4. Guţuleac E., Zaporojan S., Ţurcanu I., Furtuna A. „Analiza Performanţelor Sistemelor de Calcul prin Reţele Petri Hibride Stocastice Aproximativ Agregate", ISBN 978-9975-45-540-4, pp. 271-275, (2018).
5. A.T. Belmansour, M. Nourelfath, "An aggregation method for performance evaluation of a tandem homogenous production line with machines having multiple failure modes," Reliability Eng.and System Safety, Vol 95, No 11, pp. 1193-1201, (2010).
6. R. Bierbooms, I. Adan, M. Vuuren, "Approximate performance analysis of production lines with continuous material flows and finite buffers," Stochastic Models, 29, pp. 1–30, (2013).
7. E. Guţuleac, "Descriptive compositional HSPN modeling of computer systems," Annals of the University of Craiova, România, Series: Atomation, Computers, Electronics and Mechatronics, Vol. 3(30), No.2, pp. 82-87, (2006).
8. Rachid Hamadi and Boualem Benatallah. Dynamic restructuring of recovery nets. In Proceedings of the 16th Australasian database conference - Volume 39 (ADC '05), Hugh E. Williams and Gill Dobbie (Eds.), Vol. 39. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, pp.37-46, ISBN:1-920-68221-X, (2005).
9. E. Guţuleac, Evaluarea performanţelor sistemelor de calcul prin reţele Petri stochastice. Ed. „Tehnica-Info", Chişinău, - 276 p, (2004)
10. Hummel, Thorsten & Fengler, Wolfgang & Adamski, Marian & Karatkevich, Andrei & Wegrzyn, Marek. Design of Embedded Control Systems Using Hybrid Petri Nets. 10.1007/0-387-28327-7_12, (2005).
11. J. Dingle, Nicholas & J. Knottenbelt, William & Suto, Tamas. PIPE2: A tool for the performance evaluation of generalised stochastic Petri Nets. SIGMETRICS Performance Evaluation Review. 36. 34-39, (2009).
12. Drath, R.: Modeling Hybrid Systems Based on Modified Petri Nets. PhD Thesis, TU Ilmenau, (1999)
13. Drath, R.: Hybrid Object Nets: An Object-oriented Concept for Modeling Complex Hybrid Systems. In: Hybrid Dynamical Systems. Third International Conference on Automation of Mixed Processes, ADPM'98, Reims, (1998)
14. Alla, H., David, R., Le Bail, J.: Hybrid Petri nets. Proceedings of the European Control Conference, Grenoble, (1991)
15. The Petri Net Markup Language – http://www2.informatik.hu-berlin.de/top/pnml/.
16. D. Gasevic and V. Devedzic, "Software support for teaching Petri nets: P3," Proceedings 3rd IEEE International Conference on Advanced Technologies, Athens, Greece, pp. 300-301, doi: 10.1109/ICALT.2003.1215093, (2003).
17. WoPeD: Workflow Petri Net. Designer – http://www.woped.org/.
18. H. Verbeek and W. van der Aalst, "Woflan 2.0: A Petri-net-based workflow diagnosis tool," in Proc. 21st International Conference on Application and Theory of Petri Nets (ICATPN'00), vol. LNCS 1825. Springer, pp. 475–484, (2000).