Probabilistic and Hybrid Model Checking Deployments for Wireless Sensor Networks

MOHSIN MEMON*, SANIA BHATTI**, AND SHEERAZ MEMON***

RECEIVED ON 03.11.2011 ACCEPTED ON 01.12.2011

ABSTRACT

In the early stages of system development, model checking is a good practice for examining the WSNs (Wireless Sensor Networks). Model checking involves verifying a system's properties based on the system's finite state model. For varying applications like computers and wireless communication prior to expensive simulations, model checking has become a vital requirement in order to investigate the performance and reliability. In this paper for the first time, we are presenting probabilistic and hybrid model checking tools which are being implemented to analyse and verify the WSN applications and their examples. Here we are categorizing the model checking tools and presenting how they have been used for the investigation of various behaviours of WSN solutions. Consequently, this paper helps readers/researchers to choose the appropriate model checking tool and to get benefited in shape of validating their solutions. The paper has also highlighted the problems of existing model checking tools within WSN domain.

Key Words: **Modelling, Formal, Sensor, Network, Probabilistic Model.**

1. INTRODUCTION

Model checking is based on the concept of exhaustive reconnaissance of the system's reachable state space. It verifies the system related properties based on a finite state model of the system. Consequently, it can only be applied to the finite state space systems, which is a major limitation. For the analysis of behaviour of a system and its formal specification, one of the techniques is modelling WSNs. Modelling a real WSN includes assumptions concerned with the limitations of the network's nodes and their properties, which lead to an intricate model.

In this way, for such cases model checking helps identifying possible scenarios of the network's correctness and application related protocol initially in the system development process. Although simulation provides judgment for best-case and worst-case behaviours and explores the complex protocols, yet formal modelling and analysis serves as its alternative [1]. Protocol designers are assisted by the network simulators that measure their performance in particular circumstances and it is hard to thoroughly scrutinize likely scenarios for protocol's correctness. Simulation only keys out a single run where

* Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba, Japan.
** Assistant Professor, Department of Software Engineering, Mehran University of Engineering & Technology, Jamshoro.
*** Assistant Professor, Department of Computer Systems Engineering, Mehran University of Engineering & Technology, Jamshoro.

as model checking keys out the set of all possible runs [2]. A skewed comparison of model checking and simulations considering performance, design errors and usage memory is presented in [3]. The requirement of less CPU time, memory and less complexity for simulations in comparison to model checking is confirmed via experimental results. This evaluation can be made a base to find out the usage of model checking and simulations but this evaluation is incomplete because an adequate set of properties for simulations and model checking is not considered. This is a biased comparison because it takes into account the statement coverage metric only and this could be broaden for the consideration of other useful network related metrics.

Representation of the system's behaviour in terms of a state graph is carried out by explicit state model checking. This involves the determination of error states through some searching algorithm/technique. State graphs of the systems containing errors are usually larger than those which don't have any, thus the creation of the graphs is unexpectedly possible, as a result of which the searching mechanism can identify and describe errors even when the searching is not complete.

Model checking tools come across two well-known difficulties which are state space explosion and environmental modelling. If the model being checked has increased space state, it gives rise to the state explosion problem [4]. Several solutions [5-6] have been proposed to entertain this problem which could be applied in the domain of WSN. The state explosion problem is ameliorated by automatic iterative abstraction-refinement methodology in [5]. Despite of the fact that directed model checking makes efficient use of guided search techniques to solve state space exploration problem, a few model checking tools have utilized them. As examples, APMC [7] applies sampling of execution paths of the system to accurately approximate the probability of a specific property to abolish this problem. SLEDE [8] utilizes abstractions in a better way to eliminate state explosion. Another example is T-check [9] which uses partial order reduction in its first-depth searching procedure. CMC [4] makes use of hash compaction by storing the signature of state instead of storage of entire state within the hash table. As a result of CMC, memory requirement is reduced by a significant amount.

In environmental modelling, parts of the system which were not analyzed during the development of model are modeled, usually the aspects related to network. False error reports, associated to the network and illegal state changes which might not have aroused during the execution of the system, are minimized by the use of environment modelling [4]. One way to address environment modelling is through automated generation techniques, for example the BEG (Bandera Environment Generator) offers automated generation of environments of java program segments. The complexities of interactions that occur between the unit being analyzed and its environment are identified distinctly in real programming language [10]. Alternative way is that the library of environment models like SLEDE [8] be provided.

We can find several surveys and comparisons regarding WSN simulations [11-13] and it is believed that for the first time classification of model checking tools is being offered along with their application in WSNs. Different researchers have developed a wide variety of general purpose tools, like UPAAL, RAPTURE, KRONOS, etc. Yet, only specific tools are being implemented for verifying and analyzing WSN protocols. This gave us a direction to study and explore the performance of model checking tools in WSN solutions. One of the reasons of this ignorance is the generation of infinite state space by WSN protocols because of their scalable behaviour. The second reason is the discrepancy between an initial developed model and the real implemented WSN application, the verification model does not assure the solution's correctness [4].

This paper contributes dually. Firstly, it categorizes different model checking tools. Secondly, it presents their comparison on the basis of different parameters. This paper constitutes the discussion over classification of model checking tools with their WSN examples and a case study. In Section 2, we have discussed the various model checking tools which had been made use of, for the analysis and verification of different network scenarios. In Section 3, we present the comparative analysis of model checking tools. In Section 4, a case study related to the probabilistic model checking of a target tracking protocol is presented. And finally, we have presented the conclusion.

1.1 Probabilistic Model Checking

Probabilistic model checking is a technique intended for the analysis of probabilistic systems using quantitative properties. A mathematical model for the system is developed which is analyzed by specifying properties. States and transitions are the main components of this model. The possible configuration of the system is represented by states and evolution of the system among the states. The stochastic system behaviours are scrutinized through the construction of probabilistic models typically Markov chains or Markov processes. These models are capable of exploring the entire state space exhaustively [14]. The complexity associated with model checking techniques is polynomial in the size of the model and exponential in the size of specification. The size of the space state increases with even this complexity which arouses the state space explosion problem. The size of transition matrix is influenced by the state space explosion resulting in the inflexible model verification [7].

1.2 Hybrid Model Checking

The model checking in which the mathematical formulation of a system is involved for verifying its continuous and discrete states is referred to as hybrid model checking. It is achieved by modelling the system as a set of interacting hybrid automata. In hybrid model checking, evolution of continuous states is represented by differential equations where as the evolution of discrete states is managed by finite automation [15]. In a hybrid system, a state is described as a differential value related with the location and a value of continuous states. The inputs given to the continuous states with the differential equations associated with their location determine the progress of the system and its outputs. The system states guarantee guard conditions and applied inputs which change the transitions of states and reset at some new value of location [16].

2. MODELLING TOOLS FOR WSNS

It is believed that model checking is less complicated verification methodology than simulations. After analysis of modelling in WSN, we became aware that there are limited model checking tools for WSN domain and in most of the studies general model checking tools are used. For example, T-check [9] using TOSSIM as its base is a tool to analyze TinyOS WSN applications. SLEDE [8] is another tool specially designed for WSN security protocols. This Section comprises of details of model checking tools that are effectively used for the analysis of various WSN protocols. Classification of model checking tools is shown in Fig. 1.

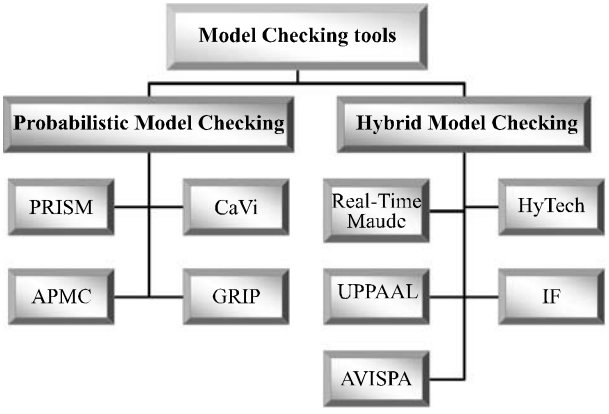

FIG. 1. CLASSIFICATION OF MODEL CHECKING TOOLS

2.1 Anquiro

It is found at the time of write up of this study that the only model checker which is intended particularly to validate WSNs is Anquiro [6]. It abstracts away low level communication aspects to avoid state space explosion problem. It employs domain specific constructs with the help of generalization point of the checking engine. The Anquiro first translates the source code into its specific language and then verify the models using LTL properties.

The model analysis of a data dissemination protocol Trickle is presented in [6]. Three abstraction levels which are specific to hardware, neighbourhood communication and system communication are defined by Anquiro. In this study a simple application is developed to verify dissemination process by executing the property to check eventual reliability of the Trickle.

2.2 APMC

APMC is an approximate probabilistic model checker that utilizes client/server architecture to build and validate a model. The GUI (Graphical User Interface) is employed to input the model along with its parameters via modified form of RM (Reactive Modules) language. The function of server is to execute the model, translate it into C source code, compile and send to the clients in reply of requested queries. APMC decreases the memory required for the verification by generating the reduced code and only one path. Conversely, the distributed nature of APMC does not permit reduction of computation time. In addition, independent path verification helps to circumvent load balancing problem. The two major components of APMC are compiler and deployer. The APMC compiler is responsible for adhoc verification, a sample generation and checking of a model and LTL property. The function of APMC deployer is to utilize the verifier and the collection of accessible computing resources in order to accumulate the approximated value of probability [14]. A comparison of dining philosopher problem modelled via PRISM with the APMC model is presented in [7]. The results confirm that APMC is able to treat with larger systems than PRISM due to the storage of one path at a time by the verification process.

2.3 PRISM

PRISM model checker is specific to analyze DTM (Discrete and Continuous-Time Markov Chains) and CTMC MDP (Markov Decision processes) and their extensions via rewards. A PRISM model consists of a set of modules which are written in RM formalism which is a form of textual modelling language. States and transitions are the two main components of a PRISM module and further quantitative measures are specified through rewards. States specify the likely behaviour of the system and alteration of behaviour is represented by transitions. The properties to scrutinize the models can be specified using Continuous Stochastic Language or Linear Temporal logic. Three different computation engines are supported by PRISM. The first engine is used for modelling of large state models and second engine is giving better performance due to the usage of sparse matrices and arrays [17]. The third hybrid engine is offering less memory usage than sparse matrix. The increase in the size of PRISM models lead towards deadlock of states which results in slow model debugging. PRISM2PROMELA [18] helps to overcome from behaviour losses by converting PRISM models into PROMELA models.

A unique probabilistic model analysis of IEEE 802.11 and S-MAC protocols via PRISM is presented by Ballarini and Miller [19]. S-MAC utilizes IEEE 802.11 randomized back-off procedure as its base with additional sleeping schedules for the nodes to facilitate energy consumption reduction. However, this model is limited to the analysis of a simple three hop topology.

2.4 CaVi

CaVi features a graphical user interface for realistic modelling of WSNs through physical parameters and its

visualization. Cavi tool can translate the inputted WSN into either a state transition model for PRISM model checker or to a format suitable for Castalia simulator, thus providing an association between formal model checking and simulation. The best-case and worst-case behaviours of the nodes deployed in the WSN are recognizable through it. The characteristics supported by CaVi can be changed for a node or for whole network including wireless medium, topologies, network and physical parameters. CaVi is a tool to import Castalia models, automatic generation of Castalia models and their simulation. It supports the exploration of topology changes related to multi-hop reception probabilities, results of Monte-Carlo simulation and model checking for multiple values [1, 20].

2.5 GRIP (Generic Representatives in PRISM)

A tool which features symmetry reduction for PRISM model checker is GRIP. GRIP provides automatic symmetry reduction for tools which are using the PRISM input language to facilitate performance improvements of models. One of the novel features of GRIP is its pre-processor functionality for PRISM specifications. Due to provision of symmetry reduction, state space of large WSN models is considerably reduced which leads to wards finite time modelling of large CTMC models. Models developed in GRIP can consists of a variety of arithmetic and boolean expressions using multiple local state variables, communication through shared global variables and multiple asymmetric modules as components of symmetric modules. The experimental comparison of five case studies using PRISM, PRISM-symm, GRIP, and optimized GRIP is discussed in [21]. It is identified that GRIP performed better than PRISM-symm in all MDP cases and gives similar results for the larger CTMC examples.

2.6 AVISPA

AVISPA is fundamentally an automatic verification tool for internet security sensitive protocols; however this tool has also been effectively applied in the formal analysis of the WSN encryption protocol (SNEP) [22]. A variety of automatic analysis techniques are integrated within the four back ends ranging from protocol falsification to abstraction-based verification methods in the present version of AVISPA. These back-ends accept the IF specifications [23] as their input. AVISPA integrates different autonomous modules to implement protocols and utilize HLPSL (High-Level Protocol Specification Language) to verify security related properties. HLPSL is a notable language to check modern industrial protocols based on its expressiveness and modularity. SNEP is analyzed in [22] by verifying two key security properties which are authenticity and confidentiality. In the first case communication between the base station and the node is analyzed intended for recovery of node secret information. In the second case, a key distribution protocol is generated to check communication of nodes for protected messages.

2.7 HyTech

A distinguished symbolic model checker for linear hybrid automated analysis of embedded systems is HyTech. The key feature of HyTech is forward reachability analysis which assists timing and safety verification features. The reachability analysis assures appropriate operation of the system and the trace generation from start to end region. A collection of linear hybrid automata and analysis commands made the input file of HyTech. HyTech supports built-in macros for reachability analysis, parametric analysis and the generation of error trajectories [24].

Coleri, et. al present lifetime analysis of a WSN using HyTech for hybrid automata modelling in [15]. In the procedure of hybrid automata, authors added a dummy state to the clock cycles of the event occurrence within the operating system. The purpose of this dummy state is to maintain synchronization of event occurrences and overheads. The simulation of a sensor node is shown by addition of an automation that corresponds to packet generated either from neighbouring sensor nodes, or the

sensing components of the application. The modelling of every single TinyOS component is revealed by Hybrid Automata which represents three states, 'wait for event', 'energy', 'wait until event completes'. The power consumption by the network considers base station connection and the distance of every sensor node from the base station.

2.8 If (Intermediate Format)

If is a well known tool for modelling of real-time systems with timed automata concept. A set of parallel processes in the IF specification make use of shared variables or message passing for interaction. A number of applications use high-level input language and integration of IF toolset for model checking, simulations and state optimizations. If notation makes use of predefined basic data types, usual type constructors and a special type called clock to measure time progress. It also allows dynamic creation and deletion of processes [23].

If toolset is used to model flooding and directed diffusion algorithms like state graphs in [25]. WSN network lifetime is measured by energy consumption comparison of two routing protocols. The timed modelling of WSN is performed by designing nodes, communication links, environment and state space.

2.9 J-Sim

J-Sim [26] uses combination of TCL and Java for performance analysis of models. It is an integrated tool for direct verifications of network protocols using J-sim simulation code. J-Sim is built upon ACA (Autonomous Component Architecture), in which an entity is called a component. A component applies its ports to communicate with other components. On the top of ACA, a packet-switched internetworking framework called INET is implemented. The WSN simulation framework is based upon both ACA and INET. It employs best-first search to reduce the size of state space. The advantage of best-first search is the quick tracing of errors than Maude linear Temporal Logic (LTL) model checker [27]. It is confirmed in [26] by performing model checking of an ARQ (Automatic Repeat Request) protocol.

A target tracking case study using the ADOV (Adhoc on Demand Distance Vector) and GPSR (Greedy Permier Stateless Routing) is presented in [2]. The study shows the batter performance of J-Sim in comparison to Ns-2 by consumption of reasonable amount of memory and comparable execution time.

2.10 Real-Time Maude

Real-Time Maude is one of the significant tools for development of general object-oriented systems compatible for modelling a network. Reachability analysis for timed search, linear temporal logic model checking and timed rewriting for simulation purposes are included in the number of analysis techniques supported by Real-Time Maude. It permits the specification, exploration of broad range of scenarios, modelling of appropriate forms of communication and testing of various behaviours on simulation tools directly because if its ability of flexible formal specification. It supports the specification of advanced systems using various data types with self-expression by equational specifications and instantaneous transitions by rewrite rules [27]. OGDC (Optimal Geographical Density Control) algorithm is discussed in [28], which is using Real-Time Maude for its formal modelling and analysis. A number of behaviours related to OGDC density control algorithm are analyzed at the abstract level by specifying informal specifications. The given results confirmed that modelling and analysis of OGDC in Maude offers more accuracy in various situations with much less effort than a simulation tool.

2.11 SLEDE

SLEDE is particularly designed to model WSN applications. It is used to extract the PROMELA models from nesC programs which is an extension of C language with addition of interfaces, modules and configurations

to model WSNs. The generation and verification process of intrusion models is automated by providing details of message structures and their sequences related to specific protocols.

The annotation language and a collection environment models enriches SLEDE and make it capable to alleviate from state explosion problem.

SLEDE is used in [8] to verify and identify imperfections of two WSN security protocols through automatic generation of intrusion models. This work is extended and presented in [29]. The partial specification of models is used to generate fault injection models automatically. A benefit of automatic generation of these fault injection models is that the user is not required to manually alter the fault injection model due to modification of the specification code.

2.12 UPPAAL

UPPAAL is one of the timed automata modelling tools which use variety of open source libraries for validation and verification. It specifies properties by a subset of CTL (computation tree logic) by specifying state formulas to correspond to individual states and path formulas to represent paths or traces of the models. UPPAAL emphasizes user-defined functions; priorities and symmetry reduction. It accepts model via its graphical user interface and performs modelling using its model checker. It also facilitates the conversion of models to a byte-code representation by performing as a compiler [30].

UPPAAL model checker is used for the modelling and verification of the LMAC protocol in [31]. The significant functionality of LMAC protocol is to investigate the collision free assignment of time slots to sensor nodes. LMAC protocol consisting of five sensor nodes is studied via non-deterministic transitions and probabilistic choices for a comprehensive set of communication of all possible connected topologies. This work is extended in [32] by adding an optimal network set up policy using probability of waiting times. It discuses that a WSN with one additional slot than the number of nodes gives improved performance using probabilistic UPPAAL for a network of four sensor nodes. A timed model of a sensor node and a base station to distinguish ice formation on a runway pavement is presented in [16]. A WSN of four nodes with linear connection is analyzed via hybrid automata model to confirm synchronization between the sensor nodes. The models focus on the evolution of states of the nodes and coordination of the events of the nodes to transmit the estimation to the base station.

3. COMPARISON OF MODEL CHECKING TOOLS

The key features supported by different model checking tools are presented in Table 1. Another way to classify the model checking tools is the consideration of their model development way and execution approach. The model checking tools which utilize specific modelling language to state models are known as symbolic model checkers. And second category is execution-driven model checkers which accomplish the code directly, rather than symbolically evaluating it. This table also indicate the provision of symmetry reduction method by the particular tool, which helps to reduce state explosion problem by identifying redundant data. It is also shown whether the scalable modelling is supported by particular tool or not. Table 2 illustrates an outline of their usage for verification of various protocols. It is indeed inexact to bring forward the contrast between the tools presented here because the papers present diverse protocols, models and modelling granularities. Thus their results cannot be really compared. However, Table 3 presents protocols' summary that are verified by several tools in terms of CPU time, maximum number of nodes used, machine and memory.

4. CASE STUDY: PROBABILISTIC MODELLING OF FAULT-TOLERANT TRGET TRACKING PROTOCOL

In this section we illustrate the analysis of FTTT (Fault-Tolerant Target Tracking) protocol which is detailed in [35]. The detailed CTMC (Continuous-Time Markov Chain Model) model of FTTT is presented in [36] with grid arrangement of sensor nodes and synchronized events. Key modules of FTTT protocol defined in PRISM are: CusterHead, SensorNode, Snetwork and target. Two cluster heads with five and 9 sensor nodes are being used during modelling. The number of messages transmitted during the execution of FTTT protocol within T time units is shown in Fig. 2 and analyzed via the following PRISM property:

TABLE 1. FEATURES SUPPORTED BY MODEL CHECKING TOOLS

Features	PRISM	APMC	CaVi	Real-Time Maude	UPPALL	AVISPA	HyTech	IF	Anquioro	J-Sim
Probabilistic	+	+	+	-	-	-	-	-	A	A
Hybrid Model Cheking	-	-	-	+	+	+	+	+	A	A
Symbolic	+	+	+	+	+	+	+	-	+	-
Execution-Driven	-	-	+	-	-	-	-	+	+	+
Property Specification	+	+	+	+	+	+	+	+	+	+
Real-Time System	+	A	+	+	+	A	+	+	-	+
Scalability	-	+	-	A	A	-	A	-	+	A
GUI	+	+	+	-	+	+	-	-	-	+
Symmetry Reduction	+	-	+	-	+	-	-	A	-	-
+ is Characteristic Supported by the Tool, - is Characteristic Not Supported by the Tool, and A is Ambiguous										

TABLE 2. OVERVIEW OF CASE STUDIES OF MODEL CHECKING TOOLS

Tools	Case Studies
PRISM	More than 45 case studies such as, computer netowrks protocols, communcation and multimedia protocols, bilogical processes, dynamic power management system, queuing system, and security protocols.
CaVI	Used for modelling realistic WSNs.
GRIP	Applied for Aspnes and Herlihy reandomized consensus protocol, randomized Byzantine protocol, Rabin's randomized mutual exclusion algorithm, FGF signalling path way, P2P protocol.
APMC	Verification of an atomic broadcast protocol, bilogical processes and mainly communication protocols.
Real-Time Maude	Formal verification and analysis of OGDC WSN algorithm, CASH scheduling algorithm, AERNCA network protocol suite, communication protocols and time dpendent cryptographic rprotocols.
UPPAAL	A number of industrial case studies such as bounded retramission protocol, collision avoidance protocol, Philips audio protocol, Leader election algorithm, consensus algorithm, Fischer's mutual exclusion algorithm, Zeroconf protocol, minimum cost forwarding protocol.
AVISPA	Library with 112 security problems derived from 33 protocols. The secrecy of a number protocols namely EKE<EKE2, IKEv2-CHILD, TLS, UMTIS-AKA, CHAPv2 present in the AVISPA library can be judged in small amount of time.
HyTech	Applied in contol-based applications including a distributed robot controller, Philips audio control protocol, generalized realroad controller, control parameters of a steam boiler, an aircraft landing-gear system and nonlinear temperature controller.
IF	Applied in Ariane-5 flight software, Medium Altitude Reconnaissance System by NLR, Sensor voting and monitoring system by IAI, ATM adaptation layer transport protocol (SSCOP), Medium access for ATM.
J-Sim	Used for verification of ARQ Protocol, AODV and directed diffusion protocols.

R{"sendReceive"}=? [C<=T] (1)

This property utilizes a reward structure named "sendReceive", which corresponds to sending receiving states of the CTMC model, representing the count of messages being sent by the cluster heads and received successfully by the sensor nodes. A reward structure called "EnergyConsumed" is added to assign the actual values of energy drainage during idle, sleep and data transmission states. The total energy consumed during the execution of FTTT protocol within T time units is shown in Fig. 3 and calculated via the property:

R{"EnergyConsumed"}=? [C<=T] (2)

Fig. 2 exhibits linear increment in the expected number of messages depending on the number of sensors in the network. Hence, as the number of sensors is increased from 5-9, the expected number of messages is also improved by the same trend. Fig. 3 shows a balanced raise in the expected total energy consumption depending on the increment of number of sensors from 5-9 during the course of time. It is observed that energy consumption is directly proportional to the number of sensor sending data to the CHs.

TABLE 3. PROTOCOLS VERIFIED BY MODEL CHECKING TOOLS

Tools	Protocol Considered	Machine Used	No. of Nodes	States	CPU Time	Memory
J-Sim [2]	Target Tracking Case Study	Dual Processor ADM Athlon, 1.gGB Memory Running Linux	22	Not Given	12000 (Seconds)	13500 (Khytes)
Anquiro [6]	Trickle	Linux PC with a P4, 3.2 GHz CPU and 2GB RAM	30	109841	562.23 (Minutes)	1282.76 (MB)
APMC [7]	Pnueli and Zuck's Dining Philosophers Algorithm	A Cluster of 20 ATHLON XP 1800+ Running Linux	300	Not Given	4071 (Seconds)	1012 (Kbytes)
Slede [8]	Needham-Schroeder Protocol	Dell Power Edge 1850 with 3.8GHz CPU and 2GB RAM	2	1647	Fraction of Sconds	3.2 (MB)
PRISM [21]	Randomized Byzanine Agreement Protocol	2.8 GHx PC with 1GB RAM	16	1.9e+15	975.5 (Seconds)	13,306,326 (MTBDD Nodes)
GRIP [21]	Randomized Byzannine Agreement Protocol	2.8 GHx PC with 1GB RAM	16	Not Given	160.1 (Seconds)	1,874,953 (MTBDD Nodes)
Real-Time Maude [28]	OGDC	3.6GHz Intel Xeon	6	Not Given	4187 (Seconds)	525 MB
UPPAAL [34]	Biomedical Sensor Network Using IEEE 802.15.4 Standard	Netebook with Celeron 1.73GHz with 1.5GB RAM	11	Not Given	315.57 (Seconds)	73.45 (MB)
AVISPA [33]	215 Security Problems Derived from 33 Protocols	PIV2.4 GHz with Linux and 1GB Memory	Not Given	Not Given	Less than 24 Minutes	Not Given

5. CONCLUSIONS

Although a well known reality is that large networks could be simulated in comparison to performing model checking, yet model checking has assisted in attaining accuracy evaluation which is otherwise complicated to attain through simulations. In this paper we have debated the standard formal modelling tools like PRISM, APMC, Real-Time Maude , UPAAL along with a few extensions of PRISM that is GRIP. These tools have been ascertained to be applicable for analyzing the performance and precision of WSNs in various studies elaborated in this paper. Use of exhaustive search to identify a system's weakness is one of the benefits served by formal modelling. Secondly, formal modelling produces best-case and worst-case behaviours of the system which makes it easy to explore detailed simulations. Thus, to scrutinize best possible performance of protocols, different techniques while trading off several parameters related to network might be explored. For the models of moderate size, it may not be possible to check their properties in a finite time length because of the fact that in the finite state representation,

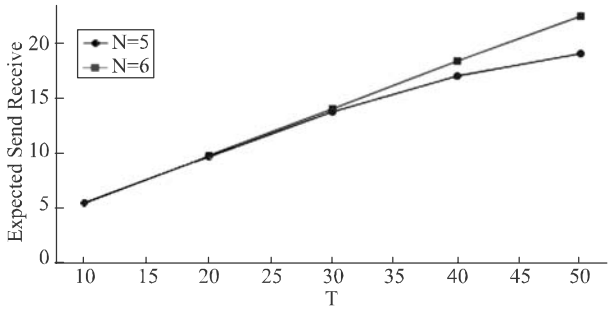

FIG. 2. NUMBER OF MESSAGES TRANSMITTED IN FTTT PROTOCOL

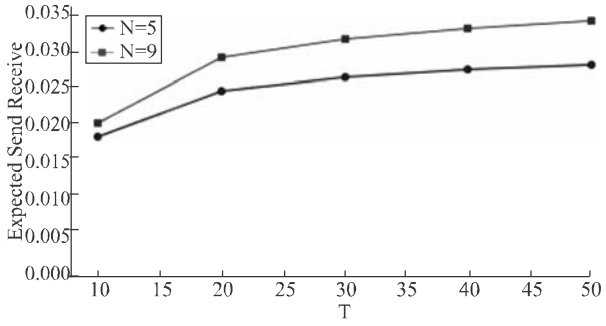

FIG. 3. TOTAL ENERGY CONSUMED BY FTTT PROTOCOL

the number of states increases in an exponential fashion with the number of variables, thus limiting the applicability of model checking for large systems. Model checking has a hazard that it operates on models only, because of which we need to derive a model from a program first which is one of the major drawbacks. The attributes of the model are difficult to study because of the generalized models, however examining a detailed abstraction may require much longer time. It would be impractical at this stage, to conclude which tool should be most suitable for WSNs because model checking tools in WSN domain still require exploration keeping in mind the modelling of further network related parameters, like scalability and state space exploration time. Although there are techniques to address the state space explosion and modelling environment, yet their optimal application in the domain of WSNs remains a key argument.

ACKNOWLEDGMENTS

Authors are thankful to Mehran University of Engineering & Technology, Jamshoro, Pakistan, for provision of facilities for this research. Special thanks to Internal and External Referees/Experts, for their valuable comments and suggestions regarding improvement of the paper.

REFERENCES

[1] Fehnker, A., Fruth, M., and McIver, A., "Graphical Modelling for Simulation and Formal Analysis of Wireless Network Protocols", Proceedings of Workshop on Methods, Models and Tools for Fault-Tolerance at the 7th International Conference on Integrated Formal Methods, pp. 80-87, 2007.

[2] Sobeih, A., Chen, W.-P., Hou, J.C., Kung, L.-C., Li, N., Lim, H., Tyan, H.-Y., and Zhang, H., "J-sim: A Simulation Environment for Wireless Sensor Networks", Proceedings of 38th Simulation Symposium, pp. 175-187, 2005.

[3] Verma, S., Lee, P., and Harris, I.G., "Error Detection Using Model Checking vs. Simulation", Eleventh Annual IEEE International High-Level Design Validation and Test Workshop, pp. 55-58, 2006.

[4] Musuvathi, M., Park, D.Y.W., Chou, A., Engler, D.R., and Dill, D.L., "CMC: A Pragmatic Approach to Model Checking Real Code", SIGOPS Operation System Review, Volume 36 (SI), pp. 75-88, 2002.

[5] Clarke, E., Grumberg, O., Jha, S., Lu, Y., and Veith, H., "Counter Example-Guided Abstraction Refinement for Symbolic Model Checking", Journal of ACM, Volume 50, No. 5, pp. 752-794, 2003.

[6] Mottola, L., Voigt, T., Österlind, F., Eriksson, J., Baresi, L., and Ghezzi, C., "Anquiro: Enabling Efficient Static Verification of Sensor Network Software", Workshop on Software Engineering for Sensor Network Applications, 2010.

[7] Herault, T., Lassaigne, R., Magniette, F. and Peyronnet, S., "Approximate Probabilistic Model Checking", Proceedings of Fifth International VMCAI'04, LNCS:2937, pp. 73-84, 2004.

[8] Hanna, Y., Rajan, H., and Zang, W., "SLEDE: A Domain Specific Framework for Sensor Network Security Protocol Implementation", Proceedings of the First ACM Conference on Wireless Network Security, pp. 109-118, Alexandria, VA, USA, March 31 - April 02, 2008.

[9] Li, P., and Regehr, J., "T-Check: Bug Finding for Sensor Networks", Proceedings of the International Conference on Information Processing in Sensor Networks, SPOTS Track, Stockholm, Sweden, April, 2010.

[10] Tkachuk, O., Dwyer, M.B., and Pasareanu, C.S., "Automated Environment Generation for Software Model Checking", Proceedings of the 18th IEEE Inter. Conference on Automated Software Engineering, pp. 116-127, 2003.

[11] Lessmann, J., Janacik, P., Lachev, l., and Orfanus, D., "Comparative Study of Wireless Simulators", Proceedings of Seventh International Conference on Networking, pp. 517-523, 2008.

[12] Merrett, G.V., White, N.M., Harris, N.R., and Al-Hashimi, B.M., "Energy-Aware Simulation for Wireless Sensor Networks", Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, pp. 64-71, 2009.

[13] Mekni, M., and Moulin, B., "A Survey on Sensor Webs Simulation Tools", Proceedings of International Conference on Sensor Technologies and Applications. (SensorComm '08), pp. 574-579, 2008.

[14] Herault, T., Lassaigne, R., and Peyronnet, S., "APMC 3.0: Approximate Verification of Discrete and Continuous Time Markov Chains", Proceedings of Third International Conference on Quantitative Evaluation of Systems, pp. 129-130, 2006.

[15] Coleri, S., Ergen, M., and Koo, T.J., "Lifetime Analysis of a Sensor Network with Hybrid Automata Modelling", Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, pp. 98-104. ACM Press, 2007.

[16] Green, J., Bhattacharyya, S., and Panja, B., "Real-Time Logic Verification of a Wireless Sensor Network", Proceedings of World Congress on Computer Science and Information Engineering, pp. 269-273, 2009.

[17] Kwiatkowska, M., Norman, G., and Parker, D., "PRISM: Probabilistic Model Checking for Performance and Reliability Analysis", ACM SIGMETRICS Performance Evaluation Review, Volume 36, No. 4, pp. 40-45, 2009.

[18] Power, C., and Miller, A., "Prism2Promela", Proceddings of Fifth International Conference on Quantitative Evaluation of Systems, pp. 79-80, 2008.

[19] Ballarini, P., and Miller, A., "Model Checking Medium Access Control for Sensor Networks", Second International Symposium on Leveraging Applications of Formal Methods, Verification and Validation, Paphos, Cyprus, 15-19 November, 2006.

[20] Boulis, F.A., Fehnker, A., Fruth, M., and McIver, A., "CaVi-Simulation and Model Checking for Wireless Sensor Networks", Proceedings of the Fifth International Conference on Quantitative Evaluation of Systems, Septmber 14-17, 2008.

[21] Donaldson, A., Miller, A., and Parker, D., "GRIP: Generic Representatives in PRISM", Proceedings of 4th International Conference on Quantitative Evaluation of Systems, pp. 115-116, IEEE CS Press, 2007.

[22] Tobarra, L., Cazorla, D., and Cuartero, F., "Formal Analysis of Sensor Network Encryption Protocol (SNEP)", Proceedings of IEEE International Conference on Mobile Adhoc and Sensor Systems, pp. 1-6, 2007.

[23] Bozga, M., Graf, S., Ober, I., Ober, I., and Sifakis, J., "The if Toolset", Corradinni, F., and Bernanrdo, M., (Editors), Proceedings of SFM'04, Volume 3185 of LNCS, Springer-Verlag, 2004.

[24] Henzinger, T.A., Ho, P.-H., and Wong-Toi, H., "Hytech: A Model Checker for Hybrid Systems", Proceedings of the 9th International Conference on Computer Aided Verification, pp. 460-463. Springer-Verlag, 1997.

[25] Mounier, L., Samper, L., and Znaidi, W., "Worst-Case Lifetime Computation of a Wireless Sensor Network by Model-Checking", Proceedings of the 4th ACM Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor and Ubiquitous Networks, Chania, Crete Island, Greece, October 22-22, 2007.

[26] Sobeih, A., Viswanathan, M., and Hou, J.C., "Check and Simulate: A Case for Incorporating Model Checking in Network Simulation", Proceedings of ACM-IEEE MEMOCODE, 2004.

[27] ¨Olveczky, P.C., and Meseguer, J., "Specification and Analysis of Real-Time Systems Using Real-Time Maude", Proceedings of FASE, Volume 2984 of LNCS, Springer, 2004.

[28] Olveczky, P.C., and Thorvaldsen, S., "Formal Modeling and Analysis of the OGDC Wireless Sensor Network Algorithm in Real-Time Maude", Proceedings of Formal Methods for Open Object-Based Distributed Systems, pp. 122-140, 2007.

[29] Hanna, Y., and Rajan, H., "Verifying Fault-Tolerance of Sensor Network Applications Using Auto-generated Fault Injection Mechanisms", Technical Report 07-11, Computer Science, Iowa State University, 2007.

[30] Behrmann, G., David, A., Larsen, K.G., Hakansson, J., Petter-son, P., Yi, W., and Hendriks, M., "Uppaal 4.0", Quantitative Evaluation of Systems, IEEE Computer Society, pp. 125-126, 2006.

[31] Fehnker, A., Hoesel, L.V., and Mader, A., "Modelling and Verification of the LMAC Protocol for Wireless Sensor Networks", Proceedings of IFM, pp. 253-272, 2007.

[32] Vighio, M.S., and Ravn, A.P., "Analysis of Collisions in Wireless Sensor Networks", 21st Nordic Workshop on Programming Theory, Copenhagen, Denmark, 2009.

[33] Vigano, L., "Automated Security Protocol Analysis with the AVISPA Tool", Electronic Notes in Theoretical Computer Science, Volume 155, pp. 61-86, 2006.

[34] Tschirner, S., Xuedong, L., and Yi, W., "Model-Based Validation of QoS Properties of Biomedical Sensor Networks", Proceedings of the 8th ACM International Conference on Embedded Software, pp. 69-78, 2008.

[35] Bhatti, S., Xu, J., and Memon, M., "Clustering and Fault Tolerance for Target Tracking using Wireless Sensor Networks", International Journal of IET Wireless Sensor Systems, Volume 1, No. 2, pp. 66-73, 2011.

[36] Bhatti, S., Memon, S., Jokhio, I.A., and Memon, M., "Modelling and Symmetry Reduction of a Target Tracking Protocol using Wireless Sensor Networks", International Journal of IET Communications, (Unpublished).