Formal Analysis of a Bio-Inspired Wireless Sensor Network Routing Algorithm

Iulisloi Zacarias[1], Cristiano Bertolini[1], Edison Pignaton de Freitas[2]

[1]Departamento de Tecnologia da Informação – Universidade Federal de Santa Maria (UFSM)
Caixa Postal 54 – 98.400-000 – Frederico Westphalen – RS – Brazil

[2]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{izacarias, cristiano.bertolini}@ufsm.br, epfreitas@inf.ufrgs.br

Abstract. *Wireless Sensor Networks (WSN) are becoming more common and applied in many areas like healthcare, environment monitoring and security. Also, it is increasing the complexity and the network interoperability as heterogeneous sensor nodes are used to compose WSN for emerging applications. Observing the importance of guaranteeing certain properties in the operation of this networks, this paper presents a formal study of a bio-inspired WSN routing algorithm using a probabilistic model to verify properties of this routing algorithm. It is noteworthy to mention the dynamic behavior of the WSN under concern, which is composed of static and mobile nodes, these last ones represented by Unmanned Aerial Vehicles (UAV). This fact adds complexity to the proposed analysis, but presents opportunities for interesting investigations. The acquired results shows that the algorithm is deadlock and livelock free, besides the evaluation of other properties that are also formally verified.*

1. Introduction

Recent advances in technology enable the manufacture of physically smaller and cheaper electronic components. This fact made possible the construction of different types of sensors, with application in many areas related to physical, chemical and biological processes [1]. Using cheaper technologies and miniaturization techniques of components, sensors have evolved acquiring reasonable processing capacity, giving them the ability to operate in different networks configurations [2]. Associate to this evolution, the enhancements in wireless communication enabled the emergence of wireless connections among sensor nodes (i.e. the deployment of Wireless Sensor Networks).

Wireless Sensor Networks (WSN) are an important technology which called the interest of researchers because it can be used in many areas, ranging from security surveillance system to detect unauthorized people, monitoring the rainfall and the environment, management of natural disasters to health care and military applications [3]. WSN differs from conventional computer networks mainly on its goal. While in a conventional network the user is interested in the computation performed by a particular node, in the WSN the main interest is the data acquired by sensor nodes [2].

This work is based on the proposal of a biologically inspired routing algorithm, used to guide the messages over the WSN. The biologically inspired approaches are a

scientific trend that emerged in the mid-1940s, but has received increased attention over the past three decades [4]. They are a fusion of nature ideas with computer techniques. Actually biologically inspired approaches in computing science can be classified on three main branches: *computing inspired by nature* that uses nature as inspiration to develop new problem solving techniques of complex problems (bio-inspired computing); *simulation and emulation of nature by means of computing* increasing our understanding of nature and insights about computer models; *computing with natural materials* that uses novel materials (in addition to silicon) to perform computation.

The use of bio-inspired approaches for solving problems in computer science is a promising direction considering the emerging number of distributed computer systems, such as WSNs. However, many biological systems may present undesirable behaviors to computer systems, compromising their determinism and performance. Considering for example, the Ant Pheromone based approaches, such as those used in network routing algorithms, it is important to verify the occurrence of undesirable situations that happen in the real biological system, such as the ant death spiral [5]. This situation happens when ants start walking around their nests spreading pheromones reinforcing a pheromone trail that do not lead them to anywhere but to a path around the nest. After some time walking around the nest without find any food source, they start dyeing due to starvation. This could be mapped to a kind of deadlock or livelock condition in a network in which a communication packet is transmitted to nodes in a cycle without reaching its destination. The starvation condition could be mapped to a time-to-live variable, for instance.

Applications of WSN are considered complex and mission-critical [2]. In this way, failures in these systems can result in property losses resulting in financial harm, or in extreme cases, loss of lives [6]. Formal methods are used in the verification processes and analysis of these critical systems and can ensure that system's security and reliability features are met according with the system project.

According to Ge *et al.* [6], verification of mission-critical systems using formal techniques, like model checking [7], have been based on several formal models as Petri Nets, Finite State Machines (FSM), *Statecharts* and SCADE. Formal verification techniques, especially verification of probabilistic models, offer a powerful and reliable approach to ensure that certain properties of complex and critical system are evaluated and guaranteed.

Baier [7] defines formal methods as "the applied mathematics for modeling and analyzing Information and Communication Technology systems" and it aims to establish system correctness with mathematical rigor. Model based verification techniques describe the system behavior in a unambiguous and mathematically precise form. These systems models are checked by algorithms that systematically explores all possible states of the model, providing a basis for a wide range of verification techniques as an exhaustive exploration of model (*Model Checking*) or experiments with a restrictive set of scenarios (*simulation*).

Probabilistic model checking consists in a technique used to verify systems which exhibit stochastic behavior and it is based in a constructed model that mimics some real software, followed by a mathematical analysis of the model in order to determine the behaviour of important properties presented by original system [8]. Unlike other verifica-

tion techniques, the model checking of probabilistic models is not only used to validate a model, but also to perform quantitative measurements of properties, such as performance and reliability [9].

This work considers the routing messages algorithm Ant Pheromone Based Routing for WSNs proposed in [2] to create a formal model representation of system behavior in order to verify alarm delivery, message costs and other related properties using PRISM language [10]. The proposed algorithm was validated with simulators in [2]. However, formal verification process can increase the reliability of the algorithm. Saleem [3] argues that simulation studies must be complemented by formal verification using mathematical models.

The PRISM language is used to represent the algorithm and check some of their properties. PRISM allows system modeling using its own high-level language, based on parallel composition of several modules. The *PRISM Model Checker* is a tool for verification of probabilistic models, used for modeling and analysis of systems that exhibit stochastic or probabilistic behavior. The tool supports various types of models, as Discrete-time Markov Chains (DTMCs), Continuous-time Markov Chains (CTMCs), Markov Decision Processes (MDPs), Probabilistic Automatas (PAs), Probabilistic Timed Automatas (PTAs) and priced PTAs. The tool has been widely used for quantitative verification and model checking of processes of many areas, from wireless networks communication protocols and quantum cryptography to biological systems and in many cases flawed or anomalous behaviour has been identified [10].

The paper is organized as follows: Section 2 introduces the WSN application scenario and the main goals in WSN research; Section 3 presents a formal representation of the *Ant Pheromone Based Routing* algorithm using the PRISM language; Section 4 presents the results of model checking process, the verification of algorithm properties and some probabilistic experiments; Section 5 presents related work about model checking applied to wireless sensor networks and conventional computer networks. Finally, Section 6 presents the conclusions and directions for future work.

2. Wireless Sensor Networks (WSN) Application Scenario

In heterogeneous WSNs, the cooperation of static and mobile sensors can provide a significant expansion of coverage area that can be monitored by the WSN, mainly due to the mobility provided by mobile sensors. However, combining these two types of sensors raise some integration problems that needed to be considered and handled. These problems are relate mainly to the form of communication used between these sensors (static and mobile). As mentioned by Freitas [2], if the control of network relies on a central node to collect data, reorganize the network and send the important collected data to a back-end information system, the scalability of the WSN can be severelly affected. Another negative aspect due to the dependence on a central node is the fact that control messages need to run over the whole network, generating a large amount of communication traffic among the nodes. This excessive communication consumes a considerable amount of node's energy resources, thus diminishing their lifetime [11]. Additionally, a fact that should be taken into consideration if a central coordinating node is used is that it represents a single point of failure what is highly undesirable. Conversely, a feature that is highly desirable in a WSN is the loose coupling among nodes. If one (or more) node(s)

is(are) damaged, the network should continue operating. However nodes must cooperate in order to complete their actions without a central management entity, i.e. they should be able to organize themselves which is refers to the concept of self-organization.

Considering these desirable requirements in the operation of WSN, a decentralized mechanism is proposed by Freitas in [2]. This mechanism is based on the concept of artificial pheromones, which is biologically inspired on the behavior of ants that act constructing and following pheromone trails when they are searching for food. The purpose of the algorithm is to guide alarms issued by static sensors until they are delivered to a mobile sensor (represented by a *Unmanned Aerial Vehicle - UAV*) equipped with a more precise sensor capable of performing a threatment confirmation or a more accurate diagnosis of the alarmed event.

This work considers this routing mechanism to create a formal model representation of this system behavior in order to verify the alarm delivery and message costs properties of this mechanism.

3. The Ant Pheromone Based Routing Proposed Model

Figure 1 shows the static nodes distribution taken as base for the formal description of the algorithm. The filled circles represent each of the static sensors nodes. The label of each circle is the node identification following the pattern nX_Y where n indicates a static node and X_Y is the Cartesian coordinates of the node, ranging from 0 to 1 on the x axis and from 0 to 2 on the y axis. Value in parentheses represents the static node pheromone level, considered for this scenario snapshot. The dashed border circles are the radio communication range or each node. Overlaps of dashed circles means that there is a communication channel among nodes. Considering this scenario and pheromone level information, an alarm issued by node $n0\_1$ should travel over nodes $n1_1$ and $n1\_2$ until their deliver to UAV on shortest path. In another possivle path, the *Alarm Agent* should follow the same path represented by UAV movement. In this model representation this choice is made in a stochastic way.

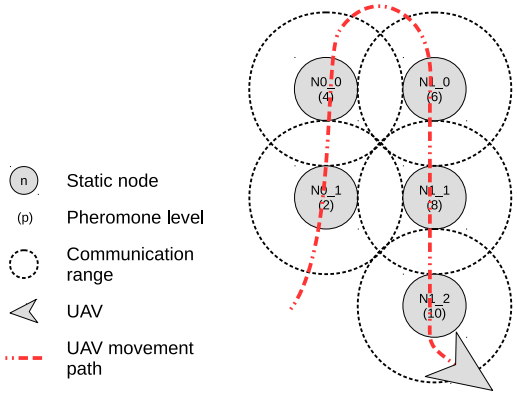

Figure 1. Distribution of the nodes in PRISM formal model.

In PRISM, a model is a set of modules and variables definition. Each component of the system is translated to a PRISM module. In this way, each static node in Figure 1 is represented by a PRISM module (e.g. $n0\_0$ represented by $module\ n0\_0\ \ldots\ endmodule$.

A PRISM module is composed by two parts, their local variables and the commands that describe the module behaviour. Local variables can be initialized by setting a value in declaration adding the keyword **init** followed by the value. PRISM supports only integer and Boolean for variable types (local or global) and it allows the use of global variables which can be accessed by all modules [12].

The state changing of a module is specified by a set of PRISM commands and reserved keywords [9]. These commands are defines as:

$$[action]\ guard -> \ probability \ : \ update;$$

where *action* is an optional synchronization label for the command and it is used to force two or more modules to make their labeled transitions simultaneously. The *guard* is a predicate over the module variables. The *probability : update* is executed only if the *guard* evaluates to *true*, which *probability* is a real-valued expression $\{probability \in \mathbb{R} \mid 0 \leq probability \leq 1\}$. The *update* is a local variable assignment (state change) [6].

According with Freitas [2], static nodes should guide the alarms through the *pheromone trail* until they are delivered to an UAV. The *pheromone trail* is created by UAV that leaves pheromones marks over the static nodes localized in area that they cross, represented by UAV movement path in Figure 1. If the nodes do not have any pheromone information the *Alarm Agent* migrates over the nodes towards a given direction that is randomly chosen by the issuer node until they reaches a pheromone trail (static node with pheromone information) or satisfies condition as number of hops or until they reaches the *Mission Area* limits. When the *Alarm Agent* reaches the mission area limit, a new direction should be chosen by the static node. In our model the number of hops were ignored because the low number of static nodes. In the algorithm proposed by Freitas [2], this direction is defined by a γ angle and may range from $[0, \pi]$. In our proposal, these directions were converted to a integer representation of directions, according with PRISM features. Therefore, the modeling of WSN only considers communications between neighbors on North, South, East and West directions and they are not able to communicate with the diagonals neighbors. As in [2], this limitation is due to the fact of low number of static nodes, otherwise each sensor node would be able to communicate with almost every other. Considering this behaviour, according to Figure 1, static node $n0\_0$ is able to communicate with the nodes $n1\_0$ and $n0\_1$, but is not able to communicate with the node $n1\_1$.

In order to simplify the model, the WSN representation is a "snapshot" of the network. Therefore, during the verification process, the UAV stays in communication with the static node $n1\_2$ all the time, during the verification process. This assumption does not affect the behavior of the model compared the real application scenario of WSN because communication between nodes occurs at a much higher speed, compared to UAV movement over the nodes.

The Listing 1 shows the definition of constants and global variables used by all modules in the PRISM model. First line defines the model type, that is a Discrete-Time Markov Chain (DTMC). In the line 2, the **prob** keyword indicates a real-valued constant $\{probability \in \mathbb{R} \mid 0 \leq probability \leq 1\}$ named $envInterference$ used to insert failure rates caused by the environment interference communication between static nodes. The $PdetectX\_Y$ and $uavInRangeX\_Y$ constants (lines 3 and 4) allow to define the alarm issuer node and the existence of a communication channel between static node nX_Y and the UAV module, respectively. The $init\_pheromone\_level\_nX\_Y$ constant allows

to configure the initial pheromone level for static node nX_Y and for each static node in the model there is a $init\_pheromone\_level$ constant. The X and Y in constant names should match the X_Y values of static node identification.

The maximum pheromone level in model is configured by the constant MAX_PH_LEVEL, and limited to "10" levels of pheromone concentration aiming to reduce the memory space used by model during model checking process.

This maximum pheromone level is due to low number of static nodes in model, allowing the delivery of *Alarm Agent* to UAV before pheromone marks expires. In fact a greater number of levels is better (Freitas[2] proposes a real-valued variable to represent pheromone marks), however it can causes the state space explosion of model. The pheromone level of nodes should be configured manually, simulating the moving of UAV over the nodes. Dashed line over the nodes in Figure 1. The model allows the configuration of pheromone decay rate of nodes, enabling or disabling it by setting this configuration using $DECAY\_RATE$ (Listing 1, line 7).

For modeling the behaviour of static nodes when it does not have any pheromone information (or pheromone information has expired) global variable is used to store the randomly chosen direction that the *Alarm Agent* should be sent (Listing 1, line 8). These directions must be represented by a set of integers (e.g. 1 means North direction, 2 means East direction, 3 means South direction and 4 means West direction).

```
1 dtmc
2 prob envInterference;
3 const PdetectX_Y = 0;
4 const uavInRangeX_Y = 0;
5 const init_pheromone_level_nX_Y = 3;
6 const MAX_PH_LEVEL = 10;
7 const bool DECAY_RATE;
8 global agentDirection: [0..4] init 0;
```

Listing 1. Probabilistic models definition, global variables and constants

Listing 2 presents the UAV module. It is simple, and goals to receive the alarm agents from static nodes. In fact, the behavior of the UAV described by Freitas [2] is much more complex. It must fly over the *Mission Area* to acknowledge the issued alarms, spread the pheromone information over the static nodes creating a *pheromone trail*, negotiate with other UAVs to check what has the most suitable sensor, according to detected threat type (assuming that there may be more than one type of threat and therefore more than one type of AgentAlarm). However, as our objective is to verify only the routing of alarms using pheromones information through static nodes, these features have been omitted.

```
10 module uav
11     uav_state: [1..4] init 1;
12     [send_nX_Y_uav] uav_state=1 & uavInRange0_0=1 -> (uav_state'=4);
13     [](uav_state=4) -> true;
14 endmodule
```

Listing 2. UAV Module

Listing 3 shows the static node module $n0\_0$. The pheromone concentration decays with the elapsed time since the static node receives the pheromone information by

UAV (*pheromone beacon*) and this behaviour is modeled by the command in line command in line 21.

The alarm delivery behavior from static node to UAV is shown in line 25, which should only occur when the UAV is on static node's communication range. Therefore, there is a communication channel between UAV and the static node (configured by setting the $uavInRangeX\_Y$ corresponding variable to "1"). When a static node has no pheromone trace ($phlvlX\_Y = 0$) it should randomly choose a direction to send the *Alarm Agent* (Listing 3, lines 28 and 29) that will follow this direction until it reaches the edge of *Mission Area* (static node has no neighbor in previously chosen direction). This situation is handled by setting the $agentDirection$ variable to value 0 which cause the node to choose a new direction (lines 31 and 32).

```
16 module n0_0
17     s0_0: [0..10] init 1; // SSTG of nodes have 10 states
18     // static node's pheromone level
19     phlvl0_0: [0..MAX_PH_LEVEL] init init_pheromone_level_n0_0;
20     // decrements the pheromone level
21     [](DECAY_RATE & phlvl0_0>0) -> (phlvl0_0'=phlvl0_0-1);
22     [](s0_0=1 & Pdetect0_0=1) -> (s0_0'=2);
23     [](s0_0=2) -> (s0_0'=3);
24     // send the alarm to UAV
25     [send_n0_0_uav](s0_0=3 & uavInRange0_0=1) -> (s0_0'=0);
26     [](s0_0=3 & uavInRange0_1=0) -> (s0_0'=4);
27     // Randomly chooses a direction to send the Alarm Agent
28     [](s0_0=4 & uavInRange0_0=0 & phlvl0_0=0 & agentDirection=0) ->
           (agentDirection'=2);
29     [](s0_0=4 & uavInRange0_0=0 & phlvl0_0=0 & agentDirection=0) ->
           (agentDirection'=3);
30     // If unfaiseable directions was choosen, reset the direction to choose new
           direction
31     [](s0_0=4 & uavInRange0_0=0 & phlvl0_0=0 & agentDirection=1) ->
           (agentDirection'=0);
32     [](s0_0=4 & uavInRange0_0=0 & phlvl0_0=0 & agentDirection=4) ->
           (agentDirection'=0);
33     // Send the alarm to the direction choosen
34     [send_n0_0_n1_0a](s0_0=4 & uavInRange0_0=0 & phlvl0_0=0 & agentDirection=2) ->
           (s0_0'=0);
35     [send_n0_0_n0_1a](s0_0=4 & uavInRange0_0=0 & phlvl0_0=0 & agentDirection=3) ->
           (s0_0'=0);
36     // If node loses pheromone level (to zero) while trying to send the alarm
37     [](s0_0>=5 & s0_0<=8 & phlvl0_0=0) -> (s0_0'=3);
38     // Send agent to all neighborhs
39     [](s0_0=4 & uavInRange0_0=0 & phlvl0_0>0) -> (s0_0'=6);
40     [send_n0_0_n1_0b](s0_0=6 & uavInRange0_0=0 & phlvl0_0>0) -> (s0_0'=7);
41     [send_n0_0_n0_1b](s0_0=7 & uavInRange0_0=0 & phlvl0_0>0) -> (s0_0'=0);
42     // Receiving messages
43     [send_n1_0_n0_0a](s0_0=0 | s0_0=1) -> 1-envInterference:(s0_0'=3) +
           envInterference:(s0_0'=s0_0);
44     [send_n1_0_n0_0b]((s0_0=0 | s0_0=1) & phlvl0_0>=phlvl1_0) ->
           1-envInterference:(s0_0'=3) + envInterference:(s0_0'=s0_0);
45     [send_n0_1_n0_0a](s0_0=0 | s0_0=1) -> 1-envInterference:(s0_0'=3) +
           envInterference:(s0_0'=s0_0);
46     [send_n0_1_n0_0b]((s0_0=0 | s0_0=1) & phlvl0_0>=phlvl0_1) ->
           1-envInterference:(s0_0'=3) + envInterference:(s0_0'=s0_0);
47     // Node fail permanently
48     [](s0_0=10) -> true;
49 endmodule
```

Listing 3. Static Node Module

The communication between static nodes were modeled by creating a command that "send" the alarm for each of neighbor nodes (the receiving node). This action syn-

chronizes the sender and receiver nodes. It was used labelling each of sender and receiver command by a identical *action label* forcing them to occur in the same time. This *action label* (in brackets at the begin of line) follows a pattern, starting by the text "send" followed by sender's node identification and the receiving node identification (lines 38 to 41). The receiving of neighbors messages are implemented with actions following the same pattern. When a node receives a message (*Alarm Agent*), it will check if the pheromone level of sender node is less or equal to their pheromone level, and will accept it. In other case (sender's node pheromone level is greater) it will discard the message. This entire model is composed by five static nodes, except node identification and neighbor's communication channels, all of them follow the same idea, behaving in a similar manner. The identification of static nodes needs to be unique in entire model and the communication channels needs to be configured according with dashed circles overlaps in Figure 1 becoming unique for each static node.

Listing 4 presents the PRISM rewards feature that computes the number of transitions related messages exchange between static nodes, or static nodes and UAV. The **reward** command is very similar to PRISM commands inside modules following the pattern:

$$[action]\ guard -> reward;$$

where *action* is an optional synchronization label, *guard* is a predicate over all variables of model and *reward* can be any expression containing variables or constants from the model. If an *action* label were used, the reward will be count only the transitions that satisfy the *guard* and are labelled with the corresponding *action* [12].

```
55 rewards "messages"
56     [send_n1_0_n0_0a] (true) : 1;
57     [send_n1_0_n0_0b] (true) : 1;
58     [send_n0_1_n0_0a] (true) : 1;
59     [send_n0_1_n0_0b] (true) : 1;
60     // ...
61     [send_n1_1_n1_2a] (true) : 1;
62     [send_n1_1_n1_2b] (true) : 1;
63 endrewards
```

Listing 4. Rewards module

4. Results

In the model checking processes, some steps should be done. Baier[7] enumerates the follow phases: Modeling phase, Running phase and Analysis phase. The description of system in PRISM language corresponds to Modeling phase and was described in the previous sections. This section will deal with the running and analysis phase which consist in running the PRISM Model Checker to verify the validity of system properties and analysis of these results.

To perform "model checking", PRISM needs first to build the probabilistic model converting the high level model description in a DTMC. In this process PRISM computes all possible states in the model, the reachable states (from the initial state) and the transition matrix. At the end of process PRISM shows information about the time taken for model construction, number of states, number of transitions, nodes in transition matrix and minterms in the log window.

In the analysis of Ant Pheromone Based Routing Algorithm the checks was performed with slightly different scenarios, by changing the issuer nodes and communication channels between static nodes and UAV to check the correctness of model.

Table 4 presentes the information about the build process. We define two possible configuration scenarios:

- Scenario 1: issuer node is $n0\_1$ and the UAV stay in communication with static node $n1\_0$;
- Scenario 2: issuer node is $n0\_0$ and the channel communication is between UAV and $n1\_2$;

As the distance between UAV and the issuer node increases, considering the pheromone trail laid by UAV following the movement path as illustrated in Figure 1, the number of states and transitions increases too. It occurs because a greater number of message exchanges (transitions) is required to deliver the *Alarm Agent* to UAV. Enabling the pheromone decay rate ($DECAY\_RATE = true$) the number of states and transitions also increases because it causes the static nodes to assume more states, one for each pheromone level.

Configuration			Model	
	DECAY_RATE	**envInterference**	**States**	**Transitions**
Scenario 1	False	0.0	16	20
Scenario 1	False	0.1	20	26
Scenario 1	True	0.0	27,336,667	174,767,048
Scenario 1	True	0.1	33,439,461	218,508,181
Scenario 2	False	0.0	25	34
Scenario 2	False	0.1	32	45
Scenario 2	True	0.0	28,980,017	184,432,062
Scenario 2	True	0.1	39,085,035	256,415,859

Table 1. Global model size

A very important property to be verified on model checking processes, even though very simple, is deadlock. Sequential programs without endless loops normally reaches a state without any outgoing transitions, called terminal state. In parallel systems these states are normally undesirable and often represents design errors. For parallel systems, deadlock indicates states in which the entire system is in a terminal state except at last one component that remains in local nonterminal state [7]. PRISM supports the checking for deadlock states using the *deadlock* keyword label which is always defined by tool which can be verified as showed in Equation 1. It results in *false* for both Scenario 1 and Scenario 2 considering an environment interference 0%.

$$E\ [\ F\ \text{“}deadlock\text{”}\ ] \tag{1}$$

One of the most important property verified in the WSN model is the deliver of *Alarm Agent* to UAV. It can be verified by PRISM by using Linear Temporal Logic (LTL) operator *F* called *eventually* or sometimes *future* [7, 12]. It means that the condition on right hand side of the *F* operator becomes true at some point along the execution path.

In Equation 2 the expression in brackets asserts that, some time, in the future, the UAV will receive the *Alarm Agent* message. The P operator allows reason about the

probability of an event's occur. In the form *P=?*, PRISM returns a real value meaning the probability of the event happening.

With the pheromone decay rate option disabled and without environment interference it always evaluate to value "1" which it means *true*. In this way, we can affirm that in optimal conditions the UAV will always receive the alarms.

$$P_{=?} \left[F \text{ “}uav\_receive\text{”} \right] \tag{2}$$

Figure 2 shows the results of the Equation 2 by inserting an ambient interference rate between 0% and 20%. The dashed line indicates the unit rate of paths (probability) in the model that will deliver the alarm to UAV (success) whereas the continuous line indicates the probability of alarm delivery to UAV by setting the pheromone level of static nodes to "0". For example, introducing an environment interference of 5%, the UAV receive probability is about 85% guiding the alarm by pheromone trail whereas the scenario without pheromone train presents a probability about 70%. It's important to point that verification does not care about the medium access control (MAC) error detection which occurs in the lower layers of network communication.

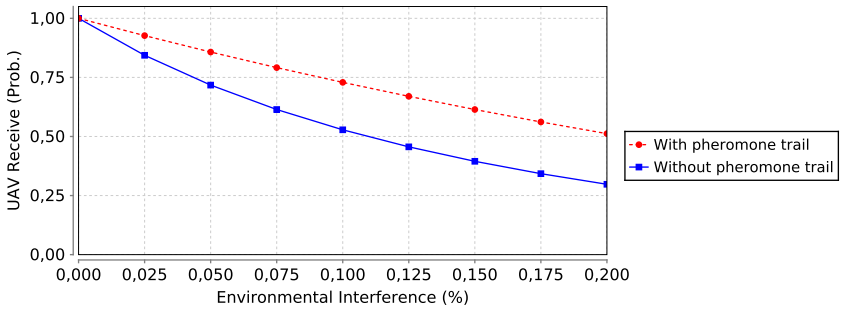

Figure 2. Environmental interference *versus* alarm receive probability by UAV with disabled pheromone decay rate feature.

Figure 3 presents the results of same verification experiment considering the pheromone decay rate on static nodes. Lower pheromone levels present a better result in this experiment. It occurs because the pheromone levels decreases in a "interleaving" way [7]. In this way, pheromone trails can be incomplete or not continuous, thus, guiding the alarms to wrong direction. In this case, the *Alarm Agent* stay "travelling" through WSN until the pheromone marks in static nodes reach the zero value, and then, static nodes will choose an aleatory direction to send the alarm towards.

The communication channel and issuer node configurations used in both experiments (Figure 2 and Figure 3) are the same as described above, for "Scenario 2" of Table 4. Issuer node is $n0\_0$ and the channel communication is between UAV and $n1\_2$ static nodes.

Another important property verified in the model relates to the fact that there is, at least one, system execution path delivering the messages to UAV module. This can be checked with Equation 3. This is a non-probabilistic property and PRISM returns their result as Boolean values. We check the model for this property by introducing an

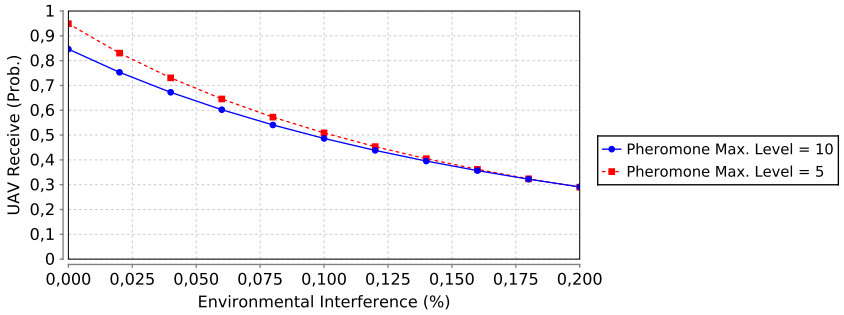

Figure 3. Environmental interference *versus* alarm receive probability by UAV considering the pheromone decay rate.

environment interference ranging from 0% to 20% and considering the pheromone decay rate both enable and disable. All results of this experiment evaluates to *True* meaning that always exists a path to deliver the *Alarm Agent* to UAV.

$$E\ [\ F\ \text{“}uav\_receive\text{”}\ ] \tag{3}$$

PRISM tool allows to increase the information about a model using the “Rewards” feature. A particular type of “Rewards” can compute the number of accumulated rewards along a execution path of model, until a certain state of system is reached [12]. Considering the way of reward compute, if the probability of system reaches the specified state is less then 1, the reward formula evaluates to infinity. Taking this into account, this property only can be checked turning off pheromone decay rate configuration, and setting the environment interference to 0% (no interference). An important state in *Ant Pheromone Based Routing* is the delivery of *Alarm Agent* issued by static node to UAV module. This state is labeled as $uav\_receive$ in PRISM model.

The Equation 4 allow us to reason about the average number of messages exchanged by static nodes, in each of execution path, until the alarm agent is delivered to UAV sensor and which returns a value of *6.5 messages* for each execution path (average).

$$R_{\{\text{“}messages\text{”}=?\}}\ [\ F\ \text{“}uav\_receive\text{”}\ ] \tag{4}$$

It is important to observe that the model is deadlock free and livelock free as pointed by PRISM verification. The alarm delivery verification properties shows highest probability of alarm delivery, even inserting environment interference, by guiding the alarm over the static nodes pheromone trail over the scenarios with no pheromone marks pointing greater efficiency of *Ant Pheromone Based Routing* algorithm.

5. Related Work

Matoušek *et al.* [13] propose an approach using formal verification techniques to analyze computer networks in order to identify potential configuration problems, defects and malfunctions that manifests only under certain specific conditions. For example, the changes in operational status of network link causes the network to operate in a different topology. Such problems are difficult to detect using traditional techniques used in network

monitoring and testing. In this paper, the authors proposes an analytical model for verification of important properties in networks computers with mutable topology. A formal model, based on graphs, representing the network topology was presented. This model allows the specification of a set of attributes used in network security analysis. The main contribution is the formal model developed, taking as base the routing and packet filtering processes. Related to our paper, the main difference is that Model Checking techniques were employed in the verification of conventional computers networks whereas in our study these techniques will be used for verification of a wireless sensor network. Despite having a different purpose, computer networks and wireless sensor networks have a common characteristic: their topologies may change constantly.

The study presented by Saleem *et al.* [3] has shown that formal modeling techniques can be applied in order to verify performance of bio-inspired routing protocols, proposing a generic framework for this type of analysis. The authors emphasize the importance of using mathematical tools for verification of such protocols, that usually employ simulation tools only. The main difficulties mentioned to the creation of this framework are the stochastic nature of the physical medium used by wireless sensor networks; continuous change of the network topology; random geographic arrangement of sensor nodes in real wireless sensor networks; the stochastic principle of "re-broadcasting" used by many routing protocols employed in this type of network; stochastic routing mechanism employed by bio-inspired protocols and finally, there is no consensus on the optimal route concept. The following performance metrics were evaluated: routing overload, optimal route and power consumption. The behavior of "BeeSensor" and "Ad hoc On-Demand Distance Vector" (AODV) protocols were modeled using the proposed framework. The development and application of these framework has led to discovery of behaviors of these protocols that could not be inferred through simulation techniques. The techniques proposed by Saleem *et al.* [3] were used as background material for the development of this work.

A study on the feasibility of the Probabilistic Model Checking was demonstrated by Duflot *et al.* [9] analysing the discovery phase used by *Bluetooth* technology. Due to the transmission characteristics, the data communication process via *Bluetooth* depends on the device initialization. This is not a trivial process in communication and so it was selected by the authors as object of study. The analysis employed probabilistic model checking techniques and, in particular, the PRISM tool. The process includes the construction of a probabilistic model based on system description and followed by the verification of system's probabilistic properties. This study take in account the properties related to the communication performance. One of the difficulties faced by the authors is related to the size of probabilistic model even applying simplification techniques. The paper presents some similarity to our proposal in the methodology employed. The main difference is on the analyzed technology, which is also based on wireless data transmission, but we are focusing on package routing across the network.

An implementation of the Tampere's hospital mobile WSN, Finland, was studied by Abo and Barkaoui [14] which applies the PRISM Model Checker to analyze the reliability and performance of the network. The main objective is to represent the behaviour of network by a formal model based on Markov chains and then check some formally properties using temporal logic specification. The Tampere's hospital wireless sensor network

enables medical personnel to send wireless alarms in threatening situations. The WSN is composed of mobiles nodes (i.e., the alarming devices continuously carried by personnel within an hospital unit), fixed routers nodes and fixed sinks (the last two devices are fixed). The authors have proposed a way to model communication between nodes, as well its mobile characteristic because nodes can move between the coverage areas of the routers. To check the communication system properties, and different nodes mobility models, the authors employed *stochastic π-calculus* allowing the change of mobility model without changing the communication model. The resulting global model was translated to Continuous-Time Markov Chain (CTMC) using the PRISM language. There are some similarities in the node mobility characteristics and communication model employed in the Tampere's hospital WSN and the *Ant Pheromone Based Routing* [2] algorithm. In latter static nodes should issue alarms, receive messages with artificial pheromones information and perform alarm messages routing based on pheromone information, so the routing mechanism differs widely.

6. Conclusion and Future Work

This paper presents a formal analysis of a bio-inspired WSN routing algorithm proposed by Freitas [2], an *Ant Pheromone Based Routing* algorithm, by means of a probabilistic model checking and PRISM. It verifies properties related to alarm delivery to mobile nodes by guided messages using artificial pheromone marks stored in static nodes of WSN. It takes an high level description of algorithm and modeled it as PRISM model in order to check mainly the alarm delivery properties. These properties shows excellent results for pheromone based routing protocol. The verified scenarios always shows a feasible system execution path resulting in a correct alarm delivery, even introducing environmental interference.

It shows that the algorithm is deadlock and livelock free. It is important to point that livelock and deadlock properties do not change their verified value by creating new scenarios with greater number of static nodes because compositional properties of processes algebra as pointed by [7], however, the number of exchanged messages needed to deliver the alarm to UAV node may change as more static nodes are added to the model. Also, the scenario without pheromone trail (e.g. pheromone marks expired in all nodes) is capable to deliver the alarms to an UAV, however the pheromone guided strategy produces better results, with a greater probability of alarm delivery. Even experiments considering environment interferences on static nodes communication channels always shows a feasible system execution path resulting in a correct alarm delivery. Different scenarios were used in verifications by changing the issuer nodes and communication channels with UAV.

The number of nodes used in this paper were limited to five nodes due to huge dimension of state space generated by system model. It turns the verification with more extent scenarios impracticable using the available hardware. Experiments were performed using an Intel Core i5-2410M CPU, 4 Gigabytes of RAM and Linux Ubuntu 14.04.3 LTS as operating system.

One future work is to model the communications between neighboring static nodes as broadcast transmission instead of several unicast transmissions as used in this paper. It could consider the refinement theory (e.g. CSP like style [15]), which allows verification of greater number of static nodes (by algebraic composition techniques). It

enables creating of more complex scenarios and the verification of complex behaviours of the algorithm. Larger scenarios permit, for example, the creation of "gaps" in WSN allowing verification of alternative system execution path for messages (or *Alarm Agents*) delivery. Increasing the number of static nodes allows verifying the system behaviour considering more than one threat types. Last, it would be interesting to implement a *flooding based* alarm routing model. It allows comparison of message delivery with the pheromone guided concept enabling to reason about alarm delivery cost presented by each of them.

References

[1] A. A. F. Loureiro, J. M. S. Nogueira, L. B. Ruiz, R. A. Mini, E. F. Nakamura, and C. M. S. Figueiredo, "Wireless Sensors Networks (in Portuguese)," in *Proceedings of the 21st Brazilian Symposium on Computer Networks (SBRC'03)*, Natal, RN, Brazil, May 2003, pp. 179–226, tutorial.

[2] E. Freitas, "Cooperative Context-Aware Setup and Performance of Surveillance Missions Using Static and Mobile Wireless Sensor Networks," Ph.D. dissertation, Halmstad University, 2011. [Online]. Available: http://hh.diva-portal.org/smash/get/diva2:450205/FULLTEXT01.pdf

[3] M. Saleem and S. A. Khayam, "A Formal Performance Modeling Framework for Bio-inspired Ad Hoc Routing Protocols Categories and Subject Descriptors," *Gecco 2008*, no. 5, pp. 103–110, 2008.

[4] L. N. de Castro, "Fundamentals of Natural Computing: An Overview," *Physics of Life Reviews*, vol. 4, no. 1, pp. 1–36, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1571064506000315

[5] Z. Meng, B. Zou, and Y. Zeng, "Considering direct interaction of artificial ant colony foraging simulation and animation," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 24, no. 1, pp. 95–107, 2012.

[6] X. Ge, R. F. Paige, and J. a. McDermid, "Analysing System Failure Behaviours with PRISM," *SSIRI-C 2010 - 4th IEEE International Conference on Secure Software Integration and Reliability Improvement Companion*, pp. 130–136, 2010.

[7] C. Baier and J.-P. Katoen, *Principles of Model Checking*. Cambridge, MA: The MIT Press, 2008. [Online]. Available: https://books.google.ca/books?id=nDQiAQAAIAAJ

[8] M. Duflot, M. Kwiatkowska, G. Norman, D. Parker, S. Peyronnet, C. Picaronny, and J. Sproston, "Practical Applications of Probabilistic Model Checking to Communication Protocols," in *Formal Methods for Industrial Critical Systems: A Survey of Applications*, S. Gnesi and T. Margaria, Eds. Wiley, 2012, pp. 133–150. [Online]. Available: http://eprints.gla.ac.uk/39594/

[9] M. Duflot, M. Kwiatkowska, G. Norman, and D. Parker, "A Formal Analysis of Bluetooth Device Discovery," *International Journal on Software Tools for Technology Transfer*, vol. 8, no. 6, pp. 621–632, 2006. [Online]. Available: http://dx.doi.org/10.1007/s10009-006-0014-x

[10] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of Probabilistic Real-Time Systems," in *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)*, ser. LNCS, G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806. Springer, 2011, pp. 585–591.

[11] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *Communications Magazine, IEEE*, vol. 40, no. 8, pp. 102–114, Aug 2002.

[12] *PRISM Manual*, 4th ed., University of Oxford, December 2014. [Online]. Available: http://www.prismmodelchecker.org/manual/Main/Welcome

[13] P. Matousek, J. Rab, O. Rysavy, and M. Sveda, "A Formal Model for Network-Wide Security Analysis," in *Engineering of Computer Based Systems, 2008. ECBS 2008. 15th Annual IEEE International Conference and Workshop on the*, March 2008, pp. 171–181.

[14] R. Abo and K. Barkaoui, "A Performability Analysis of Mobile Wireless Sensor Networks with Probabilistic Model Checking," *Wireless Advanced*, pp. 283–288, 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5983270

[15] C. Hoare, *Communicating Sequential Processes*. Prentice-Hall, 1985, 256 pages.