Model Checking Medium Access Control for Sensor Networks

Paolo Ballarini, Alice Miller
Department of Computing Science University of Glasgow, Glasgow, UK
Email: {paolo,alice}@dcs.gla.ac.uk

***Abstract*— We describe verification of S-MAC, a medium access control protocol designed for wireless sensor networks, by means of the PRISM model checker. The S-MAC protocol is built on top of the IEEE 802.11 standard for wireless *ad hoc* networks and, as such, it uses the same randomised backoff procedure as a means to avoid collision. In order to minimise energy consumption, in S-MAC, nodes are periodically put into a sleep state. Synchronisation of the sleeping schedules is necessary for the nodes to be able to communicate. Intuitively, energy saving obtained through a periodic sleep mechanism will be at the expense of performance. In previous work on S-MAC verification [25], a combination of analytical techniques and simulation has been used to confirm the correctness of this intuition for a simplified (abstract) version of the protocol in which the initial schedules coordination phase is assumed correct. We show how we have used the PRISM model checker to verify the behaviour of S-MAC and compare it to that of IEEE 802.11.**

Keywords

Verification; sensor networks; PRISM; medium access control

I. INTRODUCTION

The authors of this paper form part of the research team of a multi-site UK-based collaborative project, DIAS [1] (Design, Implementation and Adaptation of Sensor Networks). The major goal of this project is to develop methods and tools for the construction of entire environmental sensor network systems. One of the intentions of the project is to ensure that networks are optimal with respect to a global *cost function* specified by the network designer. Part of this global cost function will be energy consumption. Within the DIAS project a wireless sensor network (WSN) design is characterised with respect to four *dimensions*, namely: application, management, networking and operating system. The final goal is to adopt a co-design approach through which design solutions can be evaluated (with respect to the inherent global cost), by considering combinations of different possibilities for each of the four dimensions.

A fundamental objective of the DIAS project is to study how/what formal methods can be applied to the specification and verification of the various dimensions that characterise a WSN design.

In this paper we illustrate a modelling approach tailored to the automatic verification of communication protocols for wireless networks. Specifically we concentrate on a comparative analysis of the IEEE 802.11 protocol for wireless *ad-hoc* network and the S-MAC protocol, an 802.11 variant designed to preserve energy in WSN. We show how we have developed PRISM models of the 802.11 protocol, and two versions of the S-MAC protocol (with single and multiple sleeping schedules respectively). We then describe how model analysis can be performed through probabilistic model checking.

The paper is organised in the following way. In the next section we report on related work; in Section III an informal description of both IEEE 802.11 and S-MAC is provided; a brief introduction to probabilistic model checking is given in Section IV; in Section V the modelling approach is described and the different versions of the model considered. A summary of our contribution is contained in Section VI together with a discussion on future work.

II. RELATED WORK

There have been several attempts to verify communication protocols for wireless networks. In [12], simulation of Stochastic Petri Nets models is used to compare the effect of two different Distributed Coordination Functions (DCFs), namely *Basic Access* (BA) and *Request-To-Send/Clear-To-Send* (RTS/CTS), on system performance, in terms of throughput and waiting time. The models are very detailed and capture most aspects of the coordination protocols including the timing synchronisation procedure (used to maintain synchronisation amongst local clocks), as well as the basic backoff procedure for collision avoidance.

In [17], automatic verification through model-checking is used for the verification of the 802.11 standard. The focus here is on assessing the performance of the BA, two-way handshake, coordination function, which is achieved through verification of *soft-deadline* properties expressed in terms of the PCTL temporal logic. For that purpose the authors construct a probabilistic model (a Markov Decision Process) referring to a specific, fixed topology consisting of two sending and two destination stations.

The S-MAC protocol, introduced in [25] uses the same basic collision avoidance mechanism as IEEE 802.11, but involves a sleep state in which a station radio is switched off to preserve energy. In [25] protocol validation is achieved by a combination of analytical results and results obtained through simulation of a protocol implementation under TOSSIM [19], a software framework used to develop and simulate applications

[1]funded under the Engineering and Physical Sciences Research Council WINES programme

978-0-7695-3071-0/07 $25.00 © 2007 Crown Copyright
DOI 10.1109/ISoLA.2006.16

for WSN based on the TinyOS [13] operating system. The authors base their experiments on two different topologies (a two-hop and a multi-hop network) and show that the use of S-MAC results in a trade-off between energy saving and latency. However their models do not account for the S-MAC schedule coordination phase (a single sleep/listen schedule is assumed to be shared by all stations and the agreement/maintenance of such a schedule is not modelled). As a result it is not possible to evaluate the impact that the (likely) existence of different schedules has on the system performance.

The goal of our work is to build an accurate (probabilistic) model of S-MAC which includes the schedule coordination/maintenance behaviour. The results illustrated in [25] show that with S-MAC performance deteriorates, when compared to 802.11, even when the most optimistic scenario (a single global schedule exists) is considered. We aim to extend such analysis by allowing different schedules to co-exist throughout the network. Intuitively this co-existence should result in improving the latency for packet delivery at the cost of higher energy consumption for boundary stations (i.e. nodes belonging to several neighborhoods).

Our work has strong links to both [25] and [17]: we compare 802.11 and S-MAC performance referring to a fixed multi-hop topology (as in [25]), but rather than a using a simulative approach for protocol analysis, we apply (probabilistic) model checking verification, (as in [17]).

III. Medium Access Control for Wireless Networks

A communication protocol regulates the behaviour of communicating nodes within a concurrent environment. The Open System Interconnection (OSI) model [10] defines a layered architecture for network protocols. The *Medium Access Control* (MAC) layer, part of the data-link layer, determines which node is allowed to access the underlying physical-layer (i.e. the medium) at any given moment in time. A MAC scheme is mainly concerned with reducing the possibility of simultaneous transmissions (i.e. collisions) from taking place. The basic mechanism used for reducing the likelihood of collisions, usually referred to as Carrier Sense Multiple Access (CSMA), is that, before starting a transmission, any node should sense the medium clear for a given period.

Criteria such as the type of medium, the communication range, the communication form (e.g. radio, infrared), the number of possible nodes in the network, the required performance/reliability/security are used to classify computer networks. Generally speaking we refer to networks for which the communication medium is the ether as *wireless networks*. Wireless local area networks (WLANs) are wireless networks for which either the communication is managed by a centralised Access Point (AP) or, in the case of *ad-hoc*, nodes communicate in a peer-to-peer fashion through a distributed coordination function. Below we present the IEEE 802.11 MAC scheme for ad-hoc wireless networks.

A. MAC for Wireless LAN (802.11)

The IEEE 802.11 [1] is a family of standards which specifies a number of MAC schemes and the *Physical* (PHY) layer for WLANs. The primary MAC scheme of the standard is called *Distributed Coordination Function* (DCF). It describes a de-centralised mechanism which allows network stations to coordinate for the use of a (shared) medium in an attempt to avoid collision. The DCF is a variant of *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA) MAC [2] scheme developed for collision avoidance over a shared medium using a randomised backoff procedure. Two variants of the DCF have been defined in the standard: the *Basic Access* (BA) and the *Request-To-Send/Clear-To-Send* (RTS/CTS). Three time periods are considered to characterise them: the *DCF InterFrame Space* (DIFS), the *Short InterFrame Space* (SIFS) and the *Extended InterFrame Space* (EIFS), where $SIFS < DIFS << EIFS$.

BA: A station can start a transmission of data packets only after sensing the medium free for either a $DIFS$, if the previous transmission was successful, or for an $EIFS$ otherwise. On reception of a data packet, the destination station, after sensing the channel free for $SIFS$, sends an acknowledgment packet (ACK) back to the sender. A collision is recognised by the sending station if either: on termination of the transmission the channel is sensed occupied by another station, or if an ACK packet is not received within a given time. The main advantage of the BA scheme (also know as two-way handshake), is its simplicity. However, in this case, collision involves (large) data packets and can result in significantly deteriorated performance.

RTS/CTS: The same sensing/randomised-backoff procedure of the BA scheme is used but an additional handshake is involved using RTS and CTS control packets (as a result this scheme is referred to as a four-way handshake). After sensing that the medium is free, a station wishing to send data packets over the medium sends an RTS packet, which includes information on the duration of subsequent transmissions. On reception of an RTS, the destination (after sensing the medium free), replies with a CTS. The sender will start the transmission of actual data packets on reception of the CTS confirmation. Every neighbouring node overhearing the RTS/CTS exchange is aware of the future communication duration hence refrains from attempting to access the medium for the whole duration of the communication. Collisions can only happen between control packets, hence the cost of collision is usually smaller than with BA. Furthermore the RTS/CTS handshake eliminates collisions that are due to the so-called hidden terminal problem [5], which BA does not. On the other hand the additional handshake introduces a further latency delay and so the RTS/CTS scheme should only be used be in application whose average communication duration is large enough to justify the additional overhead.

Randomised Backoff procedure: With the DCF a transmitting station goes into backoff if the medium is not free for a sufficient length of time (either $DIFS$ or $EIFS$), no ACK arrives in time, or the frame to be transmitted is consecutive to a previous transmission. As soon as a backoff condition becomes true, the deferring station selects a *BackoffTime* composed of a random number (*backoffvalue*) of slot times, where each slot has size *aSlotTime*.

$$BackoffTime = Backoffvalue \cdot aSlotTime$$

The value of $Backoffvalue$ is a pseudo-random integer drawn from a uniform distribution over the interval $[0, CW]$, where CW is the *Contention Window* which has initial value $aCWmin$ (provided by the PHY) and takes values of ascending powers of 2 minus 1. Thus

$$CW = (aCWmin + 1) \cdot 2^{bc} - 1$$

where bc (the *BackoffCounter*) increases with the number of consecutive unsuccessful transmissions. Note that the likelihood of a longer backoff delay for repeatedly detected collisions (where bc is large) is increased.

The value of CW has an upper bound of $aCWmax$. Once this value has been reached, CW will remain at this value until it is reset (following a successful transmission attempt).

With no medium activity for a duration of $aSlotTime$ time units, $Backoffvalue$ is reduced by one. However, if the medium becomes busy the reduction of $Backoffvalue$ is frozen. The value of $Backoffvalue$ will only continue to be decremented when the channel has been sensed free for either a DIFS or EIFS period. The transmission restarts only when $Backoffvalue$ reaches zero.

B. MAC for Wireless Sensor Network (S-MAC)

A WSN consists of a set of battery powered, usually static, sensing devices communicating via a wireless network protocol. Applications include: weather conditions monitoring; moving objects detection/recognition; pollution measuring and military surveillance. Since replacing the battery of sensing devices is usually highly costly (often accessing the sensors after deployment is difficult), energy preservation is paramount for WSNs. Thus in considering aspects of a WSN, e.g. the communication protocol, routing algorithms or query processing methods, energy saving optimisation is crucial. In [25] the so-called S-MAC scheme, designed to reduce energy consumption of communicating nodes, is presented. It is based on the simple observation that for most WSN applications the sensed data streams are generated at low frequency (most of the time there is nothing to be sensed). Hence keeping nodes' radios switched on in an *idle listening* mode wastes energy. Four main sources of energy waste are identified.

1) Collision: re-transmission of corrupted transmitted packets.
2) Overhearing: listening to packets destined to other nodes.
3) Control packet overhead: sending/receiving control packets.
4) Idle listening: listening for possible traffic that is not sent.

S-MAC is designed to reduce energy waiste from each of these sources.Below we give a brief description of the main functionality of S-MAC. For a more detailed treatment the reader is referred to [25].

S-MAC is based on two distinct operative states for the sensing nodes, an energy expensive LISTEN mode where the radio of a node is switched on, and an energy saving SLEEP mode in which the radio is turned off. Each node uses a periodic listen-sleep schedule to switch between LISTEN/SLEEP operative modes. A complete LISTEN/SLEEP cycle is referred to as a *frame* and the *duty cycle* is the ratio of the listen interval to frame length which can be adapted according to the application requirements. The S-MAC scheme is concerned with two different aspects: choosing and maintaining of sleeping schedules for each node (usually referred to as coordinated sleeping) and collision avoidance.

Coordinated sleeping: Each node maintains a *schedule table* where the LISTEN/SLEEP period of each of its neighbours is recorded. When a node wants to send some data to one of its neighbours it will start the RTS/CTS protocol during the LISTEN phase of the destination node, whose details are retrieved from the schedule table. The schedule table is built in a distributed fashion, through broadcasting of SYNCH packets between neighbouring nodes. A SYNCH packet contains the sender's chosen schedule. Each node either chooses its own schedule or follows a schedule received from one of its neighbours. In the former case a node is referred to as a *synchroniser*, in the latter as a *follower*. As soon as a node picks a schedule, it broadcasts it so that all neighbours can update their table.

Co-existence of different schedules: Although the aim of co-ordinated sleeping is to synchronise neighbouring nodes on a single, shared, schedule, it is possible for nodes in a neighbourhood to have different schedules. This happens whenever a node that has announced its own schedule receives a different schedule from one of its neighbours. This may result, for example, if a neighbouring node, possibly because of collision, does not receive a previously broadcasted SYNCH in time. A node receiving two different schedules (referred to as a *boundary node*), records them in its schedule table and will adopt both schedules (i.e. it will switch to LISTEN state according to both schedules).

Collision avoidance: This is achieved through the 802.11 MAC. The RTS/CTS protocol is used to avoid collision for unicast packets, whereas a randomised carrier sense is used to prevent simultaneous transmission of broadcast packets (i.e. SYNCH). A unicast data packet follows the sequence RTC/CTS/DATA/ACK. After a successful RTS/CTS exchange, the corresponding sender and receiver will temporarily ignore their sleeping schedule until the data transmission is complete. They will then revert to SLEEP mode, until their next LISTEN mode is scheduled.

The synchronisation phase of the S-MAC protocol is illustrated in Figure 1, and the Listen period in Figure 2 and 3.

With the coordinated sleeping procedure of S-MAC, a recursive broadcast of SYNCH packets takes place. The first node choosing its own schedule will broadcast it to all of its neighbours (in the form of a SYNCH packet) which will (potentially) re-broadcast it to their respective neighbours

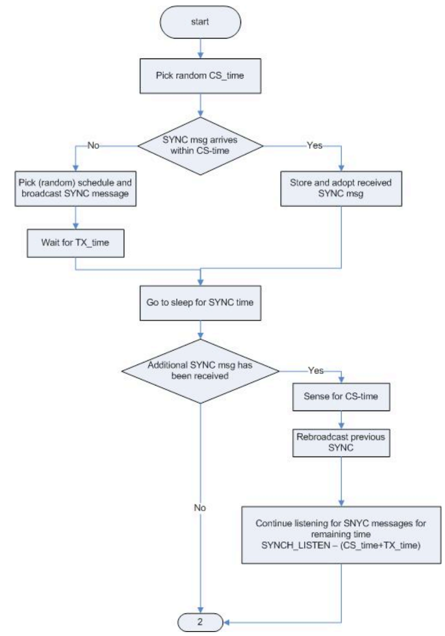

Fig. 1. Phase 1: Establishing a SYNCH table

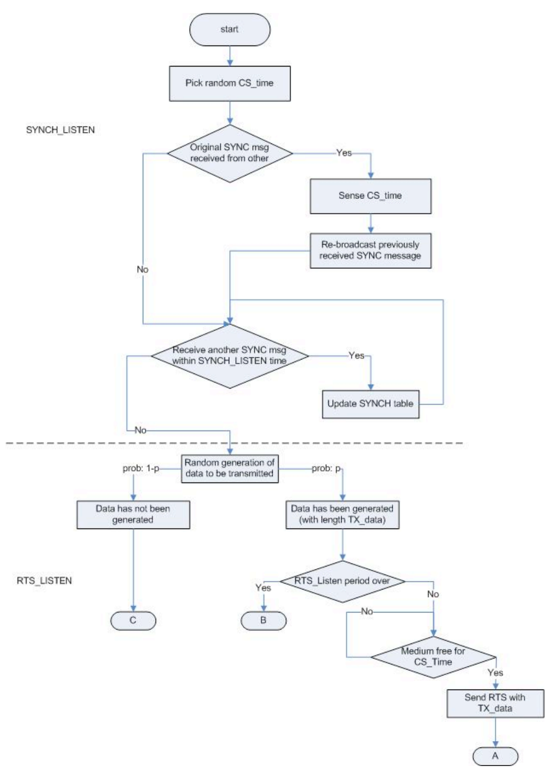

Fig. 2. Phase 2: Listen period, Upper section

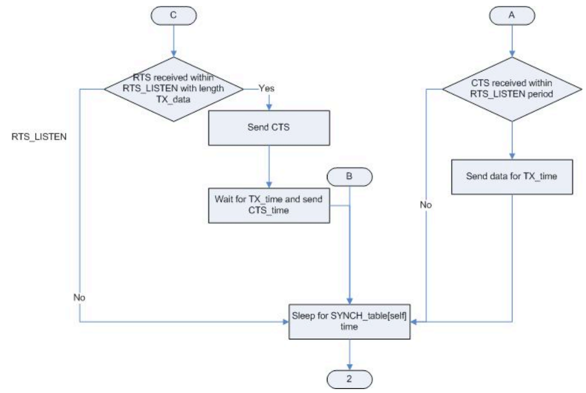

Fig. 3. Phase 2: Listen period, Lower section

(assuming that in the meantime a SYNCH packet has not been received from elsewhere). This approach may result into a flooding of a SYNCH packet throughout the network.[2].

Since all nodes shall be assumed to start the coordinated sleeping process simultaneously, no conclusion can be drawn as to which node will be first to chose its schedule and start broadcasting its SYNCH packet. Similarly it is not possible to know, a priori, how far from its originator the first SYNCH packet will reach, or equivalently, how many different schedules will co-exist in the network. The performance of S-MAC, both in terms of the induced latency and energy saving, is affected by the resulting topology of sleeping schedules. From the point of view of energy consumption, a greater number of sleep schedules should result in a higher energy consumption as boundary nodes have to wake up with respect to several schedules. From the point of view of latency, however, the existence of more schedules should improve performances, as the time for a packet to traverse a boundary node is shorter if the node is awake more often. It seems likely that a correlation would exist between the network topology and the resulting set of sleeping schedules. Nodes that are many hops away from each other are likely to adopt different schedules.

For a given network topology, the following questions are relevant:

– what is the average/maximum length (in terms of number of hops) a SYNCH packet will traverse ?
– what is the average/maximum number of different schedules a boundary node may have to follow?
– what is the average/maximum number of boundary nodes?

In this paper we use formal verification to address these questions and provide valuable information toward improved sensor network design.

IV. Verification through Model-Checking

Model checking [9], [21], [22] is a technique whereby properties of a system can be checked by building a model of the system and checking whether the model satisfies the

[2]The flooding of SYNCH packets stops as soon as each node in the network has completed its own schedule table

properties. The model is constructed using a specification language, and checked using specific model checking algorithms. Examination of counter-examples provided by the model checker enable the user to either refine the model or, if necessary, to debug the original system.

Traditional (non-probabilistic) modelling languages like Promela [15] and SMV [20], allow one to express the behaviour of a system in terms of states and transitions between states. The associated model checkers, SPIN and SMV can then be used to check qualitative properties, described using a temporal logic.

Systems that exhibit probabilistic behaviour (i.e. unpredictable processes such as, for example, fault tolerant systems or computer networks) can be verified using probabilistic model checking. A probabilistic (Markovian) model is first built and then analysed using a probabilistic model checker like PRISM [24]. In a *Discrete Time Markov Chain* (DTMC) transitions are labelled with probabilistic values in the range $[0, 1]$ and each transition is assumed to "consume" precisely one time unit. In a Continuous Time Markov Chain (CTMC) transitions are labelled with non-negative real values, time is continuous and the delay of a transition is governed by an exponentially distributed random variable with rate given by the associated real-valued label. Finally Markov Decision Processes (MDPs) are discrete-time probabilistic models that combine probabilistic behaviour with non-determinism.

Specific temporal logics have been introduced to verify probabilistic models. The logics PCTL [11] and CSL [3] are probabilistic extensions to CTL [8], and are dedicated to the verification of DTMC/MDP models and CTMC models respectively.

Both PCTL and CSL contain the probabilistic operator $\mathcal{P}$ which allows one to express properties (*probabilistic reachability properties*) concerning the likelihood of an event occurring. For example, the property: $SEND \wedge \mathcal{P}_{\geq 0.95}(trueU^{\leq 100}ACK)$ states that there is at least a 95% chance that an acknowledgment (ACK) is received within 100 time units after the transmission of a packet (SEND).

Recently Markov Reward Models (MRMs) [23] have been introduced to PRISM to enhance system verification capability. Here reward/cost values are attached to states/transition of the models. This allows one to express more detailed quantitative properties referred as *expected reachability* properties (see Section V-A for an example).

V. MODELLING 802.11 AND S-MAC IN PRISM

With PRISM we have developed DTMC models referring to a specific three-hop topology (see Figure 4) in which packets generated at one extremity (node A) reach the destination (i.e. the *sink* node D), traversing two relayers (i.e. nodes B and C). Communication range is assumed to be equal to r for each node and central nodes have two neighbours each whereas end nodes have one neighbour only.

The choice of a multi-hop topology is motivated by the need to assess the effect of coordinated sleeping on performance. With three-hop there is the possibility for (at most) two different schedules to co-exist, while keeping the model

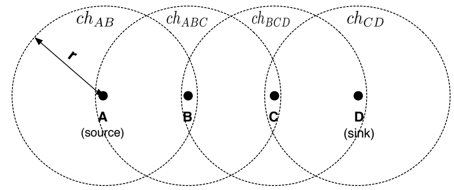

Fig. 4. A three-hop topology

size within limits. It should be noted that even though the minimal topology that allows for the possibility of having (two) different sleep schedules is a two-hops one, a topology with three-hops is to be preferred if the modeller aims to observe how communication along a route comprising more than a single relayer is affected by co-existence of different sleeping schedules.

With this topology we have modelled both the 802.11 and the S-MAC scheme. For the sake of simplicity, in the current version of each model the BA coordination function (only) has been considered for both schemes (the four-way RTS/CTS handshake significantly increases the size of the model). In the 802.11 model all nodes (apart from the source) perform idle-listening until they detect a packet on the medium, at which point they start to process communication. Observe that, when a single source node is considered, collisions between data packets can only take place at relaying nodes. In order to compare the performance of 802.11 and S-MAC we have developed two different models for the S-MAC scheme, corresponding to different levels of complexity:

– **Single (global) sleeping schedule.** All four nodes are assumed to be synchronised on a single sleeping schedule. As a result they all go to sleep and wake up at the same time. This model allows us to measure the overhead of the coordinated sleeping of S-MAC under the most simple scenario.
– **Fixed multiple sleeping schedule.** Two different schedules are assumed to co-exist, namely $schAB$ and $schCD$ where $schAB \neq schCD$. Nodes A and B follow schedule $schAB$, whereas nodes C and D go to sleep according to schedule $schCD$. For simplicity, schedules $schAB$ and $schCD$ are statically chosen. This model allows us to verify the effect of co-existence of different schedules on both latency and energy consumption at boundary nodes.

A. Model Verification

We have used the PCTL logic to express relevant properties for the verification of our models. As shown in [17] relevant properties for probabilistic model checking can be grouped in three different categories:

– Probabilistic reachability: these properties allow us to verify that relevant events take/do-not-take place at some point during the system execution. For example

$$\phi_1 \equiv \big[P\!=\!1[trueU(\text{Received\_packets} = N)]\big] \qquad (1)$$

states that "eventually (with probability at least 1) N packets will reach the sink node".

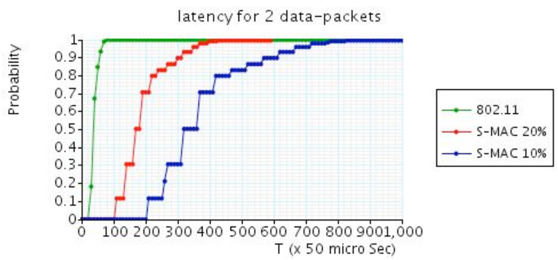

Fig. 5. Latency for 2 data-packets over 3-hops with single sleeping period

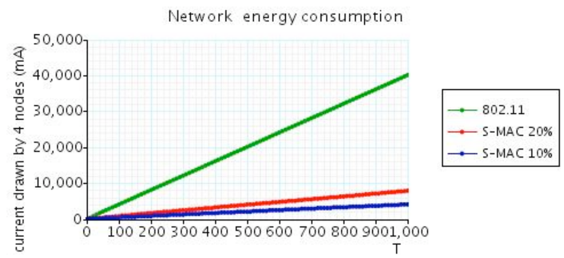

Fig. 6. Total current drawn: 802.11 vs S-MAC

– Time-bounded probabilistic reachability: these properties allow one to associate a time deadline to events that are relevant from a verification point of view. For example, for N the number of packets generated at the source node and T a discrete time boundary,

$$\phi_2 \equiv \big[P\!=?[true U(\text{Received\_packets} = N \wedge t \leq T)]\big]$$

is used to evaluate the probability that N data-packets are received at the sink node within the time deadline T.

– Expected reachability properties: these properties allow one to evaluate expected measurements of quantities that are encoded as *reward/cost values* within the model. For example by associating energy costs values to states and transitions of our model we can determine the total cost cumulated up until a given time instant T by means of

$$\phi_3 \equiv \big[R\!=?[F(time = T)]\big]$$

We first verified our models against ϕ_1 to ensure that they performed their basic functionality correctly. We then ran PRISM experiments on ϕ_2 and ϕ_3 to evaluate communication latency and energy consumption. Essentially we are interested in assessing "how long does it take for packets sent by the source node to reach the destination" (sink node) and "how much energy is consumed by the network" (expressed in terms of the sum of the current drawn by all 4 nodes in the topology) during a given time interval. In our experiments we have considered a variety of traffic loads and/or sleeping schedules. For the periodic-sleeping behaviour we have considered a fixed *listen* period equal to $250\mu s$ and we have varied the frame duration accordingly, which is: we have set it to $1250\mu s$ for a 20% duty-cycle and to $2500\mu s$ to represent a 10% duty-cycle. Figure 5 compares the latency of *pure* 802.11 (BA-CSMA without periodic sleeping) and S-MAC with, respectively, 20% and 10% duty cycle periodic sleep. It can be seen that, latency-wise, 802.11 outperforms S-MAC, with a maximum latency[3] of about $5000\mu s$, as opposed to a maximum latency of $29000\mu s$ for S-MAC 20% and of more than $50000\mu s$ for S-MAC 10%.

In Figure 6 the energy consumption for the three models is compared. In order to obtain realistic results we have used the energy consumption specifications for a MICA2 sensor node, given in Table I. Figure 6 refers to the total current consumed over a $50000\mu s$ time interval and with a traffic load of a single 1000 bits data packet. The 802.11 and S-MAC protocols are compared and it is shown that, under the considered scenario, the total current used by S-MAC with 20% and 10% duty cycle corresponds, respectively, to 19.8% and 10.4% of the total current drawn by the 802.11 protocol.

In Figure 7 and Figure 8, we compare the energy cost of S-MAC 10% and 20%. Figure 7 refers to a traffic load of a single, 500 bits packet, whereas Figure 8 refers to a 1000 bits packet (note that Figure 8 is a magnified snapshot of the S-MAC curves of Figure 9).

MICA2 sensor node specs	
Mode	Current Consumption
TX	25.0 mA
RX	10.0 mA
Sleep	0.015 mA

TABLE I
CURRENT CONSUMPTION FOR THE MICA2 NODE

It should be noted that the "step shape" of these curves is due to the presence of periodic sleeping. The flat part of each curve corresponds to the sleeping time (i.e. when the radio is turned off resulting in a much smaller energy consumption than during the "listen" period). Furthermore, in Figure 8, it should be also noted that the steps' slope decreases in time: the first 3 steps in the 20% S-MAC curve have larger slope than the last one. This reflects the fact that the maximum latency for the packet to be received at the sink node is less than $400\mu s$ (i.e. the start time for the 4^{th} "listen" time), hence from the 4^{th} period onwards the radio of each node is in receiving mode only as the only data packet considered in the experiment has, at the point, been already delivered at the sink.

Finally Figure 9 shows details of the energy consumption for S-MAC 20% with respect to a two data-packets load with variable data-packet size (200 bits, 500 bits and 1000 bits respectively). This suggests that the energy cost is not linear in the size of the packets. Specifically: the total current drawn with data-packets of 500 bits and 200 bits (i.e. $671\mu A$ and $520\mu A$ respectively) is equal to, 71.9% and the 55.7% of the current drawn (i.e. $933\mu A$) when the packets are 1000 bits each.

[3]the max latency being the point in time at which a curve reaches the value 1, ie both packets have been certainly received at the sink.

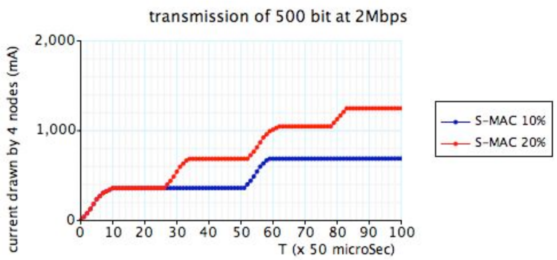

Fig. 7. Total current drawn for transmission of a 500 bits data-packet

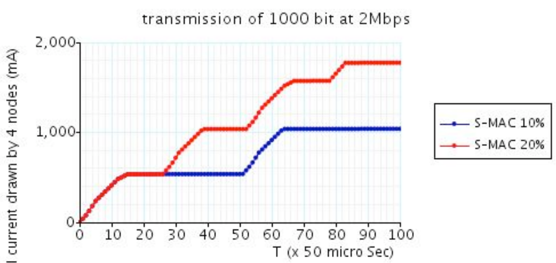

Fig. 8. Total current drawn for transmission of a 1000 bits data-packet

VI. Conclusion and Future Directions

We have described the use of probabilistic model checking for the comparative analysis of different wireless communication protocols. Specifically, we have compared the IEEE 802.11 family of protocols for (generic) *ad-hoc* networks with S-MAC, a protocol designed to reduce energy consumption suitable for WSNs. To keep our models tractable we have considered a simple 3-hop topology and developed two versions of the model aimed to evaluate protocols under different operating conditions. Model verification has been achieved through a combination of specific probabilistic reachability PCTL properties and rewards properties. Future development of this work includes completing the analysis of the described scenario to allow us to evaluate the effect of having 2 different sleeping schedules on l9atency and energy consumption. We are also planning to extend our modelling approach to other formalisms/tools including PROBMELA/LiQuor[4], [7], UPPAAL [18], and the GREATSPN tool [6] for stochastic petri nets modelling.

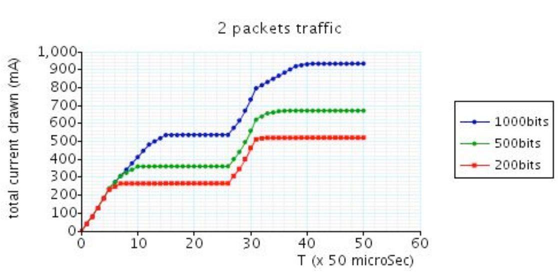

Fig. 9. Total current drawn for transmission of a 2 data-packets of different sizes

References

[1] IEEE 802.11-1997. *IEEE Wireless LAN Medium Access Control (MAC) and Physical Layer(PHY) Specification Std 802.11-1997*. Institute of Electrical and Electronic Engineers, 1997.

[2] IEEE 802.3-1985. *IEEE Wireless LAN Medium Access Control (MAC) and Physical Layer(PHY) Specification Std 802.3-1985*. Institute of Electrical and Electronic Engineers, 1985.

[3] A. Aziz, K. Sanwal, V. Singhal, and R. Brayton. Verifying continuous time Markov chains. In R. Alur and T. Henzinger, editors, *Proceedings of the 8th International Conference on Computer Aided Verification (CAV '96)*, volume 1102 of *Lecture Notes in Computer Science*, pages 269–276, New Brunswick, NJ, USA, July/August 1996. Springer-Verlag.

[4] C. Baier, M Ciesinski, and M. Grösser. PROBMELA: a modelling language for communicating probabilistic processes. In *Proceedings of the 2nd International Conference on Formal Methods and Models for Codesign*. ACM-IEEE, IEEE Computer Society, 2004.

[5] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A media access protocol for wireless LANS. In *Proceedings of SIGCOMM'94*, pages 212–225, London, UK, September 1994.

[6] G. Chiola, G. Franceschinis, R. Gaeta, and M. Ribaudo. GreatSPN 1.7: Graphical editor and analyzer for timed and stochastic petri nets. *Performance Evaluation*, 24(1–2):47–68, November 1995. Special issue on Performance Modeling Tools.

[7] F. Ciesinski and C. Baier. LiQuor: A tool for qualitative and quantitative linear time analysis of reactive systems. In *Proceedings of the 3rd International Conference on quantitative and qualitative evaluation of systems (QEST'06)*, Riverside, CA, USA, September 2006. To appear.

[8] E. Clarke, E. Emerson, and A.P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, 1986.

[9] E. Clarke, O. Grumberg, and D. Peled. *Model Checking*. The MIT Press, Cambridge, MA, 1999.

[10] J. Day and H. Zimmermann. The OSI reference model. *Proceedings of the IEEE*, 71:1334–1340, December 1983.

[11] G. Hansson and E. Macii. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(4):512–535, 1994.

[12] A. Heindl and R. German. Performance modeling of IEEE 802.11 wireless LANs with stochastic petri nets. *Performance Evaluation*, 44(1–4):139–164, April 2001.

[13] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. *SIGOPS Operating Systems Review*, 34(5):93–104, 2000.

[14] J. Hillston. *A Compositional Approach to Performance Modelling*. Distinguished Dissertations in Computer Science. Cambridge University Press, Cambridge, England, 1996.

[15] G. Holzmann. *The SPIN model checker: primer and reference manual*. Addison Wesley, Boston, 2003.

[16] Marta Kwiatkowska, Gethin Norman, and D. Parker. Probabilistic symbolic model checking with PRISM. In J.-P. Katoen and P. Stevens, editors, *Proceedings of the 8th International Conference on Tools and Algorithms for Construction and Analysis of Systems (TACAS 2002)*, volume 2280 of *Lecture Notes in Computer Science*, pages 52–66, Grenoble, France, April 2002. Springer-Verlag.

[17] Marta Kwiatkowska, Gethin Norman, and Jeremy Sproston. Probabilistic model checking of the IEEE 802.11 wireless local area network protocol. In H. Hermanns and R. Segala, editors, *Proceedings of the 2nd Joint International Workshop on Process Algebra and Probabilistic Methods, Performance Modeling and Verification (PAPM-PROBMIV 2002)*, volume 2399 of *Lecture Notes in Computer Science*, pages 169–187, Copenhagen, Denmark, July 2002. Springer-Verlag.

[18] K. Larson, P. Pettersson, and W. Yi. Model-checking for real-time systems. In Horst Reichel, editor, *Proceedings of the 10th International Symposium on the Fundamentals of Computation Theory (FCT '95)*, volume 965 of *Lecture Notes in Computer Science*, pages 62–88, Dresden, Germany, August 1995. Springer-Verlag.

[19] P. Levis, N. Lee, Welsh. M., and D. Culler. TOSSIM: Accurate and scalable simulation of entire TinyOS applications. In *Proceedings of ACM SenSys'03*, Los Angeles, CA, USA, 2003.

[20] K.L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, Boston, MA, USA, 1993.

[21] S. Merz. Model checking: A tutorial overview. In F. Cassez, C. Jard, B. Rozoy, and M. Ryan, editors, *Modeling and Verification of Parallel Processes, 4th Summer School, MOVEP 2000*, volume 2067 of *Lecture Notes in Computer Science*, pages 3–38, Nantes, France, June 2000. Springer-Verlag.

[22] M. Müller-Olm, D. Schmidt, and B Steffen. Model-checking: A tutorial introduction. In A. Cortesi and G. File, editors, *Proceedings of the 6th International Static Analysis Symposium (SAS'99)*, volume 1694 of *Lecture Notes in Computer Science (LNCS)*, pages 330–354, Venice, Italy, September 1999. Springer-Verlag.

[23] R. Smith and K. Trivedi. The analysis of computer systems using Markov reward models. In H. Takagi, editor, *Stochastic Analysis of Computer and Communications Systems*, pages 589–630. North-Holland, 1990.

[24] J. Rutten, M. Kwaitowska, G. Norman, and D. Parker. *Mathematical Techniques for Analysing Concurrent and Probabilistic Systems*, volume 23 of *CRM Monograph Series*. American Mathematical Society, Centre de Recherches Mathématiques, Université de Montréal, 2004.

[25] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *Ye, W. and Heidemann, J. and Estrin, D.*, 12(3):493–506, June 2004.