Analysis of Interrogator-tag Communication Protocols

Bojan Nokovic, Emil Sekerinski

Draft Document

December 2010

SQRL Report 60

McMaster University

This work was funded in part by NSERC Canada and by Lyngsoe Systems Canada

Abstract

In this document we discuss and analyze three different Interrogator-Tag communication protocols. The first protocol is used in the AMQM (Automatic Mail Quality Measurement) system. The second protocol is based on the ISO 18000-7 standard, which specifies the protocol and parameters for active RFID (Radio Frequency IDentification) air interface communication at the 433MHz ISM (Industrial Scientific Medical) band. The third protocol is the AMQM protocol with some features of the ISO 18000-7 standard. Quantitative properties of the protocols are analyzed. The main goal of modelling is to analyze tag message collision probability and power consumption. The model is verified by PRISM - Probabilistic Model Check Software. We showed that by implementing principles of model checking, we could verify probability of reaching a particular state, calculate collision probability as quantitative property, and cost of reaching determined state. We also showed that model of the protocol could be used to estimate possible improvement in a one protocol by implementing features from another protocol.

1 Introduction

Radio-frequency identification (RFID) is a technology that incorporates the use of an RF device (tag) applied to a product, animal, or person for the purpose of identification or tracking using radio waves. There are three types of RFID tags: active tags (with battery), passive tags (without battery), and battery assisted tags. Battery assisted tags are usually off, but when excited by some external source, like magnetic or low frequency field, the tags wake up and run on a battery.

Tags are key component of the AMQM (Automatic Mail Quality Measurement) system used by IPC (International Post Corporation) to measure the quality of mail service. The system calculates terminal dues, the fees postal operations pay to each other for delivery of cross-border mail that IPC handles for 55 of the national posts in Europe, Asia Pacific, and North America. The system measures how fast mail travels from one point to another by storing tag serial numbers and recording *time stamps* (time when message from tag is received). Collection sites are located at strategic congestion points, such as the entrance and exit gates, conveyers, sorting machines, etc. When an RFID tag enters an excitation area of 125KHz LF (Low Frequency), the tag wakes-up from *sleep* mode and transmits a preprogrammed number of messages. Each tag has a unique ID (identification). Since all tags transmitting at the same centre frequency 433.92MHz of the ISM (Industrial Scientific Medical) band, there is the possibility of message collision, when many tags transmit signal at same time.

The AMQM system is asynchronous, there is no messages acknowledge indicating that a message from a tag is successfully received at reader's side. This system is simple, but inefficient from point of view of power consumption. For instance if a tag transmits k messages and if the first message is received, $k-1$ messages will be redundant. Those messages are filtered out at the receiver. We are investigating the benefits of introducing acknowledge into AMQM protocol, similar as it is specified in ISO 18000-7 standard [6] in order to decrease number of redundant messages. ISO 18000-7 is international standard for RF tags and it is based on a internal protocol developed by Savi Technologies and specifies synchronous, two way communication.

We are exploring the possibility of representing RFID communication protocols as *probabilistic systems*. Using this representation we can analyze collision probability, probability of reaching a targeted state, and cost of reaching determined state. Collisions influences how many messages have to be sent, influences power consumption, and is crucial for the lifetime of active or battery-assisted tags.

A tag acceptance test for AMQM system is done using an experimental setup defined by IPC. Fifteen tags are sealed in mail envelopes and randomly placed into mail transportation cage. The cage is moved through area of a LF filed in which tags start transmitting pre-programmed number of messages. Transmission from each tag should be finished in four sec., and after that tag automatically goes to *sleep* mode. All messages received by RF reader are sent to a computer, which calculates the number of messages received from each tag. In order to pass the test, messages from each tag should be received in at least 96% of the time. By this setup it is possible to calculate total number of lost messages, but it is not known how many of those messages are lost because of collision and how many are lost because of other factors, such as (1) interference and (2) multi-path propagation. Interference may come from other sources, which use same frequency band. Multi-path propagation occurs when an RF signal takes different paths propagating from a tag to a RF reader. While the signal is en route, RF signal may reflect from metal surfaces and take erratic paths. In this paper we study collisions in the AMQM system by a model-based performance evaluation approach [1]. The goal is to forecast system performance and dependability, which is for AMQM system done only by testing. Collision probability is verified using probabilistic model checker tool PRISM [9]. This tool has been used for probabilistic model checking of multimedia and communication protocols *IEEE 802.11 Wireless Local Area Network Protocol* [12] and *IEEE 802.3 CSMA/CD Protocol* [10], so we believe that it is appropriate tool to be used to verify AMQM and ISO 18000-7 protocols. Some of other model checkers like ETMCC [5] and its successor MRMC[7] as well as ProbVerus [3] and Murphi [14] could also be used, but we choose PRISM because of familiarity with this tool.

2 Short Review of Markov Chain Theory

This reviews the theory from [1]. We treat *Markov Chains* (MC) from the state-based view as a graph (transition system) with probabilities, rather than the more usual interpretation as a sequence of random variables. In order to random phenomena in our modelling, transition systems are enriched with probabilities. In Discrete-Time (DT) *Markov Chains* all choices are probabilistic. We look at Markov Chains as transition systems with probability distribution for the successors of each state, so the next state is chosen probabilistically. A *Markov Decision Process* (MDP) is a generalization of a Markov Chain in which both probabilistic and nondeterministic choices coexist.

Approach

First we build a Markov model of the RFID protocol in a way presented in [1]. The model can be used to analyze *qualitative properties* and *quantitative properties*. By this approach we can analyze:

Reachability A state is called *reachable* if there exist finite execution fragment, which starts in one of initial states and ends in reachable state.

Persistence Special type of liveness properties that assert that from some moment on a certain state condition holds continuously.

Repeated reachability Can certain state be repeatedly reached ?

Optimization Find out minimal number of transmitted messages such that final state will be reached with certain probability. This is especially important when many tags (i.e., fifteen) transmit simultaneously.

Qualitative properties typically assert that event will happened with probability 1 (always) or 0 (never), while quantitative properties can have any probability (i.e., by quantitative properly we can specify that probability to reach certain state in *n* number of steps is greater than 0.92). So qualitative properties arise as a special case of quantitative properties.
In this work we will analyze *reachability* and *optimization*. Analysis of *persistence* and *repeated reachability* is out of scope of this paper.

In order to model any RFID protocol we use MC and MDP to create protocol underlying graph. MCs have been applied in various areas from biology, social sciences, psychology, to electrical engineering [8]. Markov decision process is used in stochastic control theory for study of wide range of optimization problems [15]. Model checking procedure for MC and MDP is based on Probabilistic Computation Tree Logic (PCTL) and its variant (PCTL*) which includes nondeterminism [2].

Mathematical Model

In Discrete Time Markov Chain (DTMC), successor state of state *s* is chosen according to a probability distribution. Nondeterministic choices are refined by probabilistic ones. This probability distribution only depends on the current state, and not on the path fragment that led to the state (from some initial state). The system does not depend on the history but only on the current state and this is known as *memoryless property*. A Discrete Time Markov Chain is a tuple:

$$\mathscr{M} = (S, P, l_{init}, AP, L)$$

where

- S is countable nonempty set of states
- $P : S \times S \rightarrow [0,1]$ is the transition probability function such that for all states s:
$$\sum_{s' \in S} P(s,s') = 1$$
- $l_{init} : S \rightarrow [0,1]$ is the initial distribution such that
$$\sum_{s' \in S} l_{init}(s) = 1$$
- AP is a set of atomic propositions and $L : S \rightarrow 2^{AP}$ is a labeling function.

The transition probability function P specifies for each state s the probability $P(s,s')$ of moving form s to s' in one step, i.e., by a single transition. The value $l_{init}(s)$ specifies the probability that the system evolution starts in state s. The states s with $l_{init} > 0$ are considered possible initial states. In similar way states s' for which $P(s,s') > 0$ are viewed as (possible) successors of s.

For state s and $T \subseteq S$, let $P(s,T)$ denote the probability of moving from s to some state $t \in T$ in a single step. That is,

$$P(s,T) = \sum_{t \in T} P(s,t)$$

We can identify the transition probability function $P : S \times S \rightarrow [0,1]$ with the matrix $(P(s,t))_{s,t \in S}$. The row $P(s,.)$ for state s in this matrix contains the probabilities of moving from s to its successors, while the column $P(.,s)$ for state s specifies the probability of entering state s from any other state.

The initial distribution l_{init} can be viewed as a vector $(l_{init}(s)), s \in S$. Labelling function L relates a set $L(s) \in 2^{AP}$ of atomic propositions to any state s, and 2^{AP} denotes the power set of AP.

A Markov chain induces an underlying diagraph where states act as vertices and there is an edge from s to s' if and only if $P(s,s') > 0$. Paths in MCs are minimal (i.e., infinite) paths in the underlying diagraph. They are defined as infinite state sequences $\pi = s_0 s_1 s_2 \cdots \in S^{\omega}$ such that $P(s_i, s_{i+1}) > 0$ for all $i \geq 0$. For path π in $\mathscr{M}$, $inf(\pi)$ denotes set of all states that are visited infinitely often in π. For finite Markov chains, $inf(\pi)$ is nonempty for all paths π.

Markov Chain Representation of Simple AMQM System

A tag of an AMQM system can be represented by Markov chain with only three states, as shown in Figure 1. It represents a simple, unidirectional communication protocol.

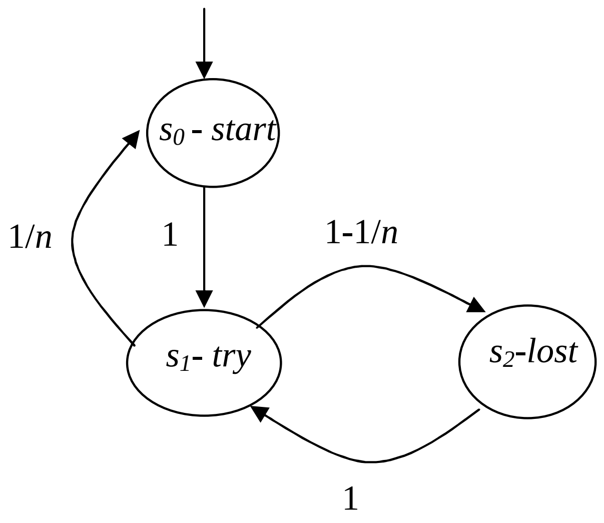

Figure 1: Markov chain of simple AMQM communication protocol

The states are $s_0(start), s_1(try), s_2(lost)$.

$$S = \{s_0, s_1, s_2\}$$

Message transmission is randomized in time and approximation of collision probability is given by formula

$$(1 - 1/n) \tag{1}$$

where n is number of tags, and $n \in \mathbb{N}_+$. We assume that all tags start to transmit at same time and transmission is done for fixed period (usually 40 messages in 4 sec.). Collision probability formula 1 is approximation of next observations

- when $n = 1$ collision probability is 0 (only one tag transmits)
- when $n = 2$ collision probability is 1/2
- when $n = 3$ collision probability is 2/3
- when $n \to \infty$ collision probability is 1

The transition probability matrix for this system, and initial distribution vector are

$$P = \begin{pmatrix} 0 & 1 & 0 \\ 1-1/n & 0 & 1/n \\ 0 & 1 & 0 \end{pmatrix} \qquad I_{init} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}.$$

An example of a path is

$$\pi = (start, try, start, try, lost, try, start)^{\omega}.$$

Infinite repetition is denoted by Greek word ω, for instance infinite repetition of *a,b,a,b...* is denoted by $(a,b)^{\omega}$. Each message along path π has to be retransmitted before delivery. For $T = \{start, lost\}$ we have $P(try, T) = 1$. So using this simple model we can verify reachability property. By adding the cost to each state, we can calculate cost (i.e., power consumption) of reaching state s_2 from state s_0. This requires extension of Markov chains, called *Markov reward* chains. A model based on a *Markov reward* chains is called *Markov reward model* (MRM). It is a tuple $(\mathscr{M}, rew)$, where $rew : S \to \mathbb{N}$ is reward function which assigns to each state $s \in S$ a non-negative integer reward $rew(s)$.

3 Communication Protocol Based on ISO 18000-7 Standard

ISO/IEC 18000-7 is intended to address RFID devices operating in the 433MHz ISM frequency band, providing an air interface implementation for wireless, non-contact information system equipment for item management applications. The RFID system includes a host system and RFID equipment. The host system runs an (application) program, which controls interfaces with the RFID equipment. The RFID equipment is composed of two principal components: tags and interrogators (or readers). The tag is intended for attachment to an item, which a user wishes to manage. It is capable of storing a tag serial number and other data regarding the tag or item and of communicating this information to the interrogator. The main differences to the AMQM protocol are:

- Messages are transmitted in time periods called *slots*.
- An acknowledge that has to be received for each message.

The interrogator is a device, which communicates to tags in its RF communication range. The interrogator controls the protocol, reads information from the tag, directs the tag to store data in some cases, and ensures message delivery and validity. This system uses active tags. Typical application is RFID asset management. RFID systems defined by this standard provide the following minimum features:

- Identifying tag in range
- Reading data and error detecting
- Writing data or handling read only systems gracefully
- Selection by group or address
- Graceful handling of multiple tags in the field of view

The tag *collection process* is used to identify tags in the operating field of the interrogator. This is an iterative process that includes methods for coordinating responses from the tag population and handling collisions which occur when multiple tags transmit at the same time. The entire tag collection process is referred to as a *complete collection sequence*.

Tag Collection

Figure 2 is an adaptation of [6] and shows a complete collection sequence consisting of a wakeup period (WP) followed by a series of collection periods (CP). Each collection period consists of a synchronization period (SP), a listen period (LP), and an acknowledge period (AP).

- Wakeup Period (WP) is the time period in which the interrogator transmits one or more Wake Up Signals to bring all tags to the ready state. The Wakeup Period is transmitted only once during a collection sequence.

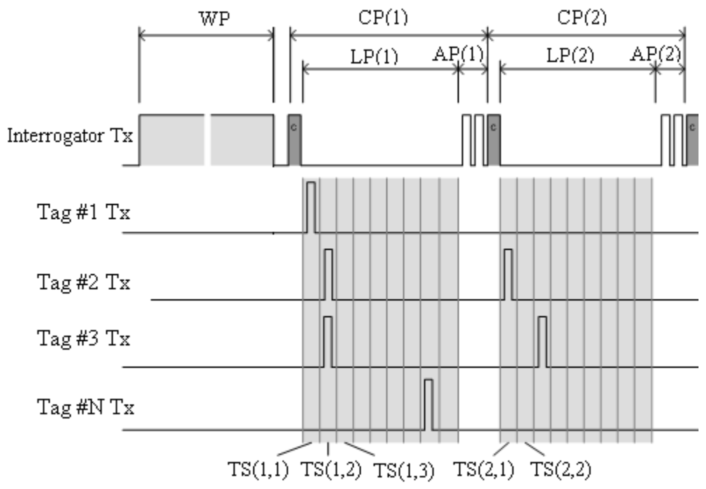

Figure 2: Interrogator-tag communication timing diagram

- Collection Period (CP) are time periods in which the tags are identified and acknowledged. A sequence of *collection periods* is used, repeating until all responding tags have been identified. Each *collection period* consists of a *synchronization*, a *listen*, and an *acknowledge* period.
- Synchronization Period (SP) is the time period in which the interrogator sends a broadcast collection command to the tag population in the operating field of the interrogator. Each tag synchronizes its timing with the end of the packet reception interrogator broadcast command.
- Listen Period (LP) is time period in which the interrogator waits for responses from tags. The listen period is divided into Time Slots (TS) which are time windows for tags to respond. Each tag selects a random Time Slot for its response, and delays its response to fit into the chosen Time Slot.
- Acknowledge Period (AP) is time period in which the interrogator acknowledges responding tags and may optionally retrieve additional data from a tag. For each tag identified by the interrogator during the previous Listen Period, the reader optionally collects additional data from the tag, and then commands the tag to sleep using the Sleep command.

Tag Collection Model

Model of communication protocol based on ISO 18000-7 standard is shown in Figure 3. There are four states: $s_0(start)$, $s_1(try)$, $s_2(lost)$, $s_3(delivered)$.

$$S = \{s_0, s_1, s_2, s_3\}$$

This protocol is synchronous, and for each transmitted message the acknowledge should be received, otherwise the message is considered lost.
For n tags ($1 \leq n$), t time slots ($1 \leq t$) and $n \leq t$, we can calculate collisions probability by calculating permutations with and permutation without repetition. From elementary permutation theory [13] we know that total number of permutation with repetition is

$$t^n \tag{2}$$

and number of permutation without repetition (collision) is

$$\frac{t!}{(t-n)!} \tag{3}$$

So, the probability β that the message will be delivered is

$$\beta = \frac{\frac{t!}{(t-n)!}}{t^n}$$

and the probability of collision α, is simply

$$\alpha = 1 - \beta.$$

Underlying Markov chain of the protocol is shown at Figure 3. The transition probability matrix and the initial distribution vector for this system are

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \alpha & \beta \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \qquad I_{init} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

The approximate collision probability shown in Figure 4. is calculated only for cases when number of *time slots* is not less than *number of tags*. It is clear that collision probability is decreasing by increasing number of time slots for the same number of tags. But, by increasing number of time slots, Collection Period (CP) becomes longer. So based on particular implementation, just looking at chart

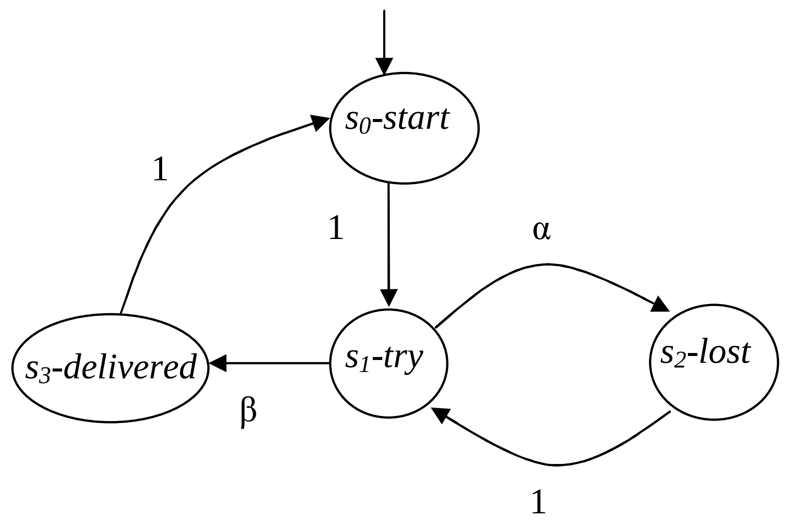
$$\begin{tikzcd}
 & s_0\text{-start} \arrow[d, "1"] & \\
s_3\text{-delivered} \arrow[ur, bend left, "1"] & s_1\text{-try} \arrow[l, "\beta"'] \arrow[r, bend left, "\alpha"] & s_2\text{-lost} \arrow[l, bend left, "1"]
\end{tikzcd}$$

Figure 3: Markov chain of communication protocol

from Figure 4) we can specify number of time slots in order to configure system for required performance. (i.e., if we have three tags in system, and requirement that collision probability is less than 50%, we need six time slots)

4 Improved AMQM Communication Protocol

We will consider an AMQM protocol with an added message acknowledge feature, and call it *improved* AMQM protocol. Tag-interrogator communication, as any wireless communication is error-prone and the messages may be lost because of interference from environment and multi-path propagation.

In our model we have a unique initial *start* state $l_{init}(start) = 1$ and $l_{init}(s) = 0$ for $s \neq start$. In the state *start*, the message is generated that is sent of along the channel in its unique successor state *try*. The message is lost with probability $1/n$, where n is number of tags, in which case the message will be sent off again until it is eventually delivered. As soon as the message is delivered correctly, the system returns to the initial state. The diagram of this communication protocol is the same as the one from Figure 3., with only difference that $\beta = 1/n$.

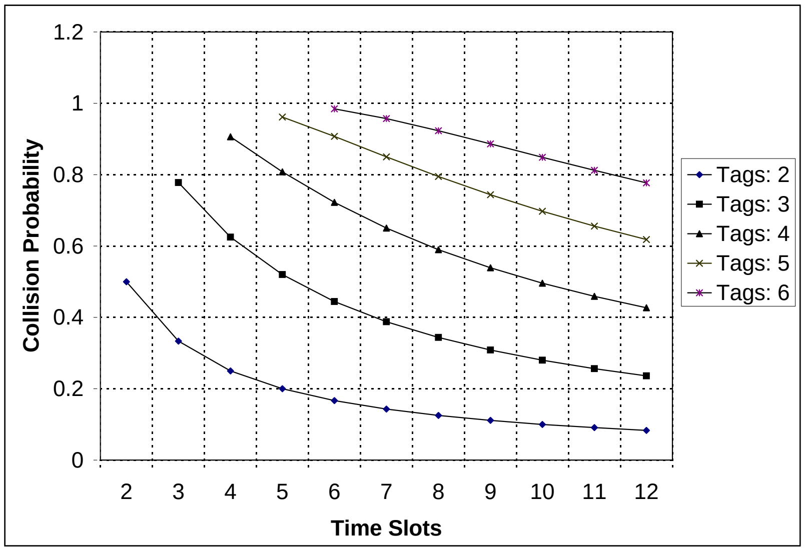

Figure 4: Collision probability as function of time slots for two to six tags

The transition probability matrix and and the initial distribution vector are

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1-1/n & 1/n \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \qquad I_{init} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

An example of a path is

$$\pi = (start, try, lost, try, delivered)^{\omega},$$

so along this path message has to be retransmitted two times before delivery. It follows that $inf(\pi) = S$. For $T = \{s_2, s_3\}$, we have $P(s_1, T) = 1$.

Computing Reachability Property

One of the elementary questions of quantitative analysis of systems modeled by Markov chains is to compute the probability of reaching a certain set of states in particular number of tries.

That set, say B, may represent set of *bad* states which should be visited only with some small probability, and set of *good* states which should rather be visited frequently. Event of interest is $\Diamond B$ (event to eventually reach some state in B) [3], and $B \subseteq S$, where S is set of all states.

An example of a path is

$$\pi_k = (start, try, (lost, try)^k, delivered)$$

where k is an arbitrary number.

In our case, we can consider event $\Diamond B$ for $B = \{delivered\}$. Probability that message will be successfully delivered is $\beta = 1/n$, $n \in \mathbb{N}_+$, and collision probability is $\alpha = 1 - \beta = 1 - 1/n = \frac{n-1}{n}$. We can calculate probability of recaching state *delivered* by infinite series

$$P_r^{\mathscr{M}}(\Diamond delivered) = \sum_{k=0}^{k=\infty} \alpha^k \cdot \beta = \frac{\beta}{1-\alpha} = \frac{\frac{1}{n}}{1-\frac{n-1}{n}} = 1.$$

If we take infinitely many steps, we will successfully reach state *delivered*, but collision probability α should be less than one. ($0 \leq \alpha < 1$)

Example:

For three tags, $n = 3$, $\beta = \frac{1}{3}$ and $\alpha = \frac{2}{3}$. The probability that the message will be delivered in three trials is sum of π_0, π_1,and π_2 which yields

$$\frac{1}{3} + \frac{1}{3} \cdot \frac{2}{3} + \frac{1}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} = \frac{19}{27} \approx 0.703$$

■

This example shows how the probability of reaching a certain set of states can be calculated by means of infinite sums, but probability of reaching a cetrian set of states B can be calculated in more efficient way, without infinite sum. Let x_s denote probability of reaching B from s, for arbitrary $s \in S$. The goal is to compute x_s for all states s. $B = \{delivered\}$ and $x_s > 0$ for all states s, since *delivered* is reachable from all states. Let $\tilde{S}$ be set of states $s \in S \setminus B$ such that there is a path fragment $s_0, s_1 ... s_n$, $(n > 0)$ with $s_0 = s$ and $s_n \in B$. For the vector $x = (x_s)$, $s \in S$ we have

$$x = Ax + b$$

where the matrix A contains the transition probabilities for the states in $\tilde{S}$, and the vector $b = (b_s)_{s \in \tilde{S}}$ contains the probabilities of reaching B from $\tilde{S}$ in one step. In our model, $\tilde{S} = \{start, try, lost\}$ and we obtain equations

$$\begin{aligned}
x_{start} &= x_{try} \\
x_{try} &= \tfrac{1}{n} \cdot x_{lost} + \tfrac{n-1}{n} \\
x_{lost} &= x_{try}
\end{aligned}$$

Those equations can be written as

$$\begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -\frac{1}{n} & 0 \\ 0 & -1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \mathrm{x}_{start} \\ \mathrm{x}_{try} \\ \mathrm{x}_{lost} \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{n-1}{n} \\ 0 \end{pmatrix}$$

which yields the unique solution $x_{start} = x_{try} = x_{lost} = 1$. Thus, the event of eventually reaching the state *delivered* is sure for any state.

This technique yields the following two-phase algorithm to compute reachability probabilities in finite Markov chains. First we perform a graph analysis to compute the set $\tilde{S}$ of all states that can reach B. After that, we generate the matrix A and vector b and solve linear equation system $(I-A)\cdot x = b$, where I is identity matrix of same rank as matrix A.

PRISM Model

Model of the improved AMQM system is built using PRISM probabilistic model checker, a tool for formal modelling and analysis of systems that exhibit random or probabilistic behavior. It supports three types of probabilistic models: discrete-time Markov chains, continuous-time Markov chains and Markov decision processes, plus extensions of these models with costs and rewards. Goal is to find out after how many cycles all messages from a bunch of 15 tags will be successfully sent off. This setup is according to acceptance requirement test for postal tag. We found that it will take up to eight transmissions from some tags before all messages are successfully received.

In this model, 15 tags are excited and all of them start transmitting messages at same time, but at different time intervals. In the first cycle no messages will be successfully sent off, but in the second cycle we will have three messages delivered and three tags (which successfully sent off messages) are shut down. We assume that acknowledge is not error-prone because it is sent from only one interrogator, so there is no collision between acknowledge messages and in this model we do not consider possible impact from environment, which can cause message lost. The process is continued until all messages are transmitted.

In *improved* AMQM system, total number of transmitted messages is

$$15+15+11+10+9+9+8+7=84 \tag{4}$$

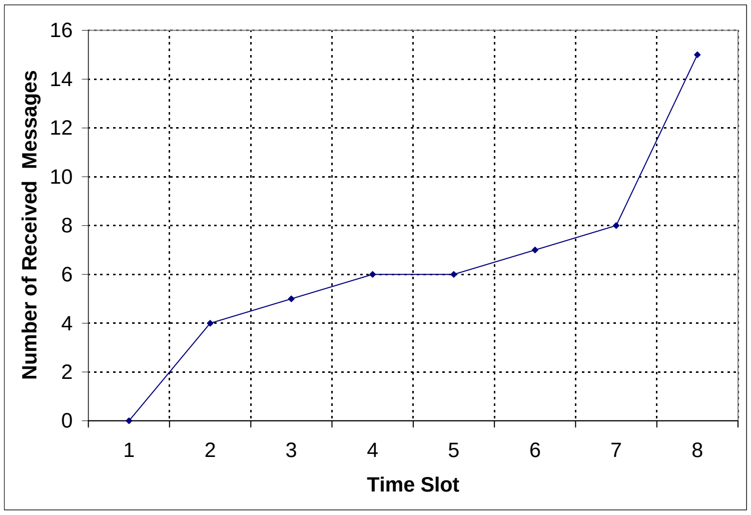

Figure 5: Number of tag messages received in each slot

In *simple* AMQM system (without acknowledge), 15 tags always transmits 40 messages each, so total number of transmitted messages is

$$15 \cdot 40 = 600 \tag{5}$$

Based on this model we can conclude that by implementing message acknowledge, number of transmitted messages may be decreased by

$$600/84 \approx 7.14 \tag{6}$$

times. If we assume that power consumption per each transmission is the same, we can conclude that adding message acknowledge into protocol will decrease power consumption by approximately seven times and consequently increase seven times lifetime of the tags, because active tag uses battery as source of power. In this approximation we assumed that power consumption during sleep mode is insignificant, but for more accurate estimation this has to be taken into consideration, so in reality calculation given by equation 6, is upper bound in possible power saving. From Fig. 5 we can see that randomization of messages transmitted by tags has significant effect only if number of tags in system is less than eight.

5 Results

We demonstrated that model-based performance evaluation could be used to analyze a simple communication protocols, like ISO 18000-7 and the AMQM protocols. Three main results of our analysis are:

- By implementing principles of model checking we verified that probability of reaching *good* state in all three communication protocols (*simple* AMQM, *improved* AMQM and ISO 18000-7) is 1. So messages will be always eventually received.
- We approximated the collision probability function and showed it as quantitative property (Figure 4) which depends of number of tags and number of time slots.
- We modified the AMQM protocol by introducing acknowledge. For this asynchronous protocol we build a model using the PRISM model checker and calculated that in eight transmit cycles we should be able to receive messages from all fifteen tags.

The first result can be used as verification that all three communication protocols are correct from point of view of reaching good state.
The second result can be used to optimize number of tags and time slots in communication system based on ISO 18000-7 standard.
The third result can be used to justify implementation of message acknowledge feature in simple AMQM communication protocol. According to our calculation, number of transmitted messages will be decreased for more than seven times. Since most of power is consumed during transmission, according to our model, we need approximately seven time less power, and that will increase tag (battery) lifetime also for approximately seven times.

Tags used in AMQM system have a lifetime of ten years, and the lifetime of a battery is five years. So, after five years battery on each tag has to be replaced. Implementation of message acknowledge in the AMQM protocol will eliminate the need to replace the battery in tag during it's lifetime. Every year approximately 20000 batteries on tags used by IPC have to be replaced, so a longer battery lifetime means that there is no need for this replacement.

6 Conclusions

Markov Chain in state-based form as graph with probabilities is suitable for modelling simple RFID communication protocols. All wireless communications are subject to external interference, so we believe that a probabilistic approach is appropriate to model wireless protocols in general.

Based on such model we can mathematically verify reachability, and persistence of any state of the protocol. Markov chains with rewards (*Markov rewards*), are chains in which states are augmented by rewards, and could be used to calculate power consumption for battery-powered systems (tags).

Our experience shows that the probabilistic model checker PRISM is sutable tool for analyzing quantitative properties of simple wireless protocols. In general, modelling is useful in the early stages of software design, when crucial design decisions can be justified by modelling results [4].

Further decrease in tag power consumption may be achieved by dynamically changing clock frequency of micro-controller. Some of those techniques are described in [16].

Environment related sources of errors such as interference and multipath propagation were not part of our modelling because of their stochastic nature.That may be studied in some our future works.

7 Appendix

PRISM Language

This is brief explanation of PRISM syntax, what the tool does, and how it does based on [11]. The main components of the PRISM language are modules and variables. The behaviour of each module is described by a set of commands. A command takes the form:

```
[] guard -> prob_1 : update_1 + ... + prob_n : update_n;
```

The guard is a predicate over all the variables in the model (including those belonging to other modules). Each update describes a transition which the module can make if the guard is true. A transition is specified by giving the new values of the variables in the module, possibly as a function of other variables. Each update is also assigned a probability (or in some cases a rate) which will be assigned to the corresponding transition

In our model, there are only two probabilities: (1) probability of collision and (2) probability to successfully deliver message. All 15 tags have the same probabilities, and we tried in maximum of 40 cycles to transmit all messages. When particular tag successfully transmits message, the collision probability decreases for $1/n$, where n number of the tags.

When all tags successfully transmits the message, the simulation will stop, and it will indicate *deadlock*, because no tag is transmitting. Our goal is to find out in how many cycles all tags will successfully deliver messages and shuts down.

Code of PRISM Model

```
//bounded retransmission
//after the message is received the tag is shut down

dtmc

//probability to get acknowledge after the first message
//transmitted is  1/N

const int N=15;

module tags
//i - number of transmission cycles
 i : [0..40] init 1;

// number of tags
// t=0 - ACK not come yet
// t=1 - ACK received
 t1 : [0..1];
 t2 : [0..1];
 t3 : [0..1];
 t4 : [0..1];
 t5 : [0..1];
 t6 : [0..1];
 t7 : [0..1];
 t8 : [0..1];
 t9 : [0..1];
 t10 : [0..1];
 t11 : [0..1];
 t12 : [0..1];
 t13 : [0..1];
 t14 : [0..1];
 t15 : [0..1];

//If ACK received, tag will shut down, and probability of received me
//other tags will be increased

[] (t_1=0 & i <= N ) -> i/N : (t1'=1) + (N-i)/N : (t1'=0) ;
[] (t_2=0 & i <= N ) -> i/N : (t2'=1) + (N-i)/N : (t2'=0) ;
[] (t_3=0 & i <= N ) -> i/N : (t3'=1) + (N-i)/N : (t3'=0) ;
[] (t_4=0 & i <= N ) -> i/N : (t4'=1) + (N-i)/N : (t4'=0) ;
[] (t_5=0 & i <= N ) -> i/N : (t5'=1) + (N-i)/N : (t5'=0) ;
[] (t_6=0 & i <= N ) -> i/N : (t6'=1) + (N-i)/N : (t6'=0) ;
[] (t_7=0 & i <= N ) -> i/N : (t7'=1) + (N-i)/N : (t7'=0) ;
[] (t_8=0 & i <= N ) -> i/N : (t8'=1) + (N-i)/N : (t8'=0) ;
[] (t_9=0 & i <= N ) -> i/N : (t9'=1) + (N-i)/N : (t9'=0) ;
[] (t_10=0 & i <= N ) -> i/N : (t10'=1) + (N-i)/N : (t10'=0) ;
[] (t_11=0 & i <= N ) -> i/N : (t11'=1) + (N-i)/N : (t11'=0) ;
[] (t_12=0 & i <= N ) -> i/N : (t12'=1) + (N-i)/N : (t12'=0) ;
[] (t_13=0 & i <= N ) -> i/N : (t13'=1) + (N-i)/N : (t13'=0) ;
[] (t_14=0 & i <= N ) -> i/N : (t14'=1) + (N-i)/N : (t14'=0) ;
[] (t_15=0 & i <= N ) -> i/N : (t15'=1) + (N-i)/N : (t15'=0) ;

//If number of cycles is less than 2/3, probably after each
//cycle is increased by 1/N, because of randomized message
//transmission form the tag
```

```
 [] (i<N/2 ) -> (i'=i+1);

//If number of cycles is more than 2/3 , probably after each cycle
//is  increased by 2/N

[] (i>2*N/3 & i<N ) -> (i'=i+2);

//maximum probability is N/N = 1

[] (i>N )  -> (i'=N);

endmodule
```

References

[1] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. The MIT Press, London UK, first edition, 2008.

[2] A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. *In 15th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS), volume 1026 of Lecture Notes in Computer Science*, 1995.

[3] V. Hartonas Garmhausen, S. Campos, and E. M. Clarke. Probverus:probabilistic symbolic model checking. in 5th international amast workshop on formal methods for real-time and probabilistic systems (arts). *Volume 1601 of Lecture Notes in Computer Science, pages 96-110 Sringer-Verlag*, 1999.

[4] P. Hawrylak, J. Cain, and M. Mickle. Analytic modelling methodology for analysis of energy consumption for ISO 18000-7 RFID networks. *Radio Frequency Identification Technology and Applications*, Vol. 1, No. 4, 2007.

[5] Holger Hermanns, Joost-Pieter Katoen, Joachim Meyer-Kayser, and Markus Siegle. A tool for model-checking markov chains. *International Journal on Software Tools for Technology Transfer, 4(2):153-172, 2003*, 2003.

[6] ISO/IEC. 18000-7 information technology, RFID for item management part 7: Parameters for active air interface communications at 433 mhz. *ISO/IEC*, 2008.

[7] Joost-Pieter Katoen, M. Khattri, and I. S. Zapreev. A markov reward model checker. *IEEE Computer Society, 243–244*, 200.

[8] V. Kulkarni. *Modeling and Analysis of Stochastic Systems*. Chapman and Hall, 1995.

[9] IM. Kwiatkowska, G. Norman, and D. Parker. Probabilistic symbolic model checking with prism: A hybrid approach. *International Journal on Software Tools for Technology Transfer*, 2004.

[10] M. Kwiatkowska, G. Norman, J. Sproston, and F. Wang. Symbolic model checking for probabilistic timed automata. *Proc. Joint Conference on Formal Modelling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault Tolerant Systems (FORMATS/FTRTFT'04)*, 2004.

[11] Marta Kwiatkowska. Prism probbailistic symbolic model checker. Website, 2010. `http://www.prismmodelchecker.org/`.

[12] Marta Kwiatkowska, Gethin Norman, and Jeremy Sproston. Probabilistic model checking of the IEEE 802.11 wireless local area network protocol. *Lecture Notes in Computer Science, pages 169-187, Springer, 2002*, 2002.

[13] Bona Milkos. *Combinatorics of Permutations*. CRC Press, 2004.

[14] G. Della Penna, B. Intrigila, I. Melatti, E. Tronci, and M. Venturini Zilli. Finite horizon analysis of markov chains with the murphi verifier. *Journal of Software Tools and Technology Transfer*, 2006.

[15] M. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, 1994.

[16] Y. N. Srikant and Priti Shankar. *The Compiler Design Handbook: Optimizations and Machine Code Generation*. CRC Press, 2007.