The IT University of Copenhagen

Preliminary Proceedings First International Workshop on Formal Methods for Wireless Systems

FMWS'08
Toronto, Canada
23 August 2008

Jens Chr. Godskesen (IT University of Copenhagen)
Massimo Merro (University of Verona)
editors

IT University Technical Report Series TR-2008-109

ISSN 1600–6100 August 2008

ISSN 1600–6100

ISBN 978-87-7949-183-0

Copies may be obtained by contacting:

IT University of Copenhagen
Rued Langgaards Vej 7
DK-2300 Copenhagen S
Denmark

Telephone: +45 72 18 50 00
Telefax: +45 72 18 50 01
Web www.itu.dk

Contents

Preface

The FMWS workshops aim at bringing together researchers interested in formal methods for wireless systems, more specifically in theories for semantics, logics, and verification techniques for wireless systems. Wireless systems are rapidly increasing their success in real-world applications while formal methods for modelling, analysing, and verifying the systems are lacking behind. Recently however much attention has been carried out to model, analyse and verify Sensor Networks and, more generally, Ad Hoc Networks.

This very first FMWS workshop is co-located with CONCUR '08, held in Toronto. The workshop contains two invited presentations, by Ansgar Fehnker and Holger Hermanns, five regular papers, and one short paper. The selected papers and abstracts for the invited talks appear in these preliminary proceedings.

We would like to thank the authors of the submitted papers, the invited speakers, the members of the program committee, and their subreferees for their contribution to both the meeting and this volume. Also we thank the CONCUR organising committee for hosting FMWS '08, and Richard Trefler for the local organization of the workshops. The final proceedings will become available electronically at Elsevier's web site `http://www.elsevier.com/locate/entcs`.

The editors

Jens Chr. Godskesen (IT University of Copenhagen)
Massimo Merro (University of Verona)

FMWS 2008 Program Committee

Jens Chr. Godskesen (co-chair), Denmark
Andrew D. Gordon, Cambridge, UK
Radha Jagadeesan, Chicago, USA
Kim G. Larsen, Aalborg, Denmark
Massimo Merro (co-chair), Verona, Italy
Sebastian Nanz, Denmark
Catuscia Palamidessi, France
Davide Sangiorgi, Bologna, Italy
Scott A. Smolka, USA
Luca Vigan, Verona, Italy

Formal Methods in the Wireless Network Domain

Ansgar Fehnker

National ICT Australia[1]
Locked Bag 6016
The University of NSW
Sydney NSW 1466
Australia

Formal Methods have a long track record of developing tools, methods and techniques, to verify protocols, software and hardware systems. Wireless Networks combine these areas in a characteristic way. Due to the nature of the network nodes, hardware is comparatively small as is the software running on it. Due to the nature of the network algorithms and protocols should be distributed and concurrent. An important aspect of a wireless networks is that nodes use multi-hop communication on an unreliable medium, and that the network is subject to dynamic changes and environmental interference. This talk presents experience from a research project on Formal Methods in the wireless network domain, and in particular how model checking can help with the design of the different aspects of wireless networks, and how they can position themselves with respect to network simulators, the main tool for developers other than testing in this realm.

The project Formal Methods of Performance Analysis of Wireless Network Applications (PEWNA) was a small-scale three year project within the Formal Methods program of National ICT Australia. The PEWNA project demonstrated the use of formal methods, in particular model checking, for wireless applications, and it applied them successfully to time-synchronization protocol, gossiping protocols, and power management protocols.

The purpose of model checking however differed for the different applications. Model checking was of course used to show correctness [4]. But for protocols that are known to fail for some problematic configurations, it can be also used to identify all problematic scenarios [5]. The counterexamples then help to address some of the problematic scenarios, even though complete correctness will not be achieved. Another use of model checking it to use for optimization to compute optimal op-

[1] National ICT Australia is funded through the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian Research Council.

eration schedules [6]. Finally, we used model checking tool to obtain performance and network measures, such a reception rates, for given protocols [1,2,3].

Given the different purposes, there are roughly speaking two use cases for formal methods. The most traditional and most common use case is to use them to thoroughly analyze a protocol or system [3,4,5,6]. For this it is necessary to develop detailed models, and use the full range of tools and techniques to tackle the problem. This use case assumes an expert user. Any improvements can be leveraged to any implementation of the protocol, which justifies the effort. On the other hand there is the case when model checking is used in the design process to compute performance measures for many iterations of a wireless network design [1,2]. Each time the designer makes a change to the topology or the network parameters, the analysis has to be repeated. To become feasible, we need abstract modelling templates such that the response time of the model checker remain acceptable. The aim is that model checking will become transparent to the user, and just one tool among others to assess the quality of the current network design. This use case targets non-experts in formal methods, with a background in wireless network design.

The main motivation at the beginning of the PEWNA project was the observation that simulation, the main tool used in the wireless network domain, is fraught with problems. It has for example been observed in that different simulators can produce vastly different results, even for very simple protocols. The size of networks that can be simulated are typically several orders of magnitude higher than the size of networks that can be model checked. However, this is a skewed comparison, since simulation, especially in the presence of probabilistic protocols and a probabilistic environment, cannot provide complete coverage. The strength of simulation to provide very detailed illustrative traces for debugging. Furthermore, model checking can be used to provide performance measures that are difficult if not impossible to obtain by simulation. The project showed that for formal methods to be successful in the wireless network domain, it is not necessary to compete with simulation tools, but that they can position themselves as valuable tools in their own right.

References

[1] Boulis, A., A. Fehnker, M. Fruth and A. McIver, *Cavi – simulation and model checking for wireless sensor networks*, in: *Quantitative Evaluation of Systems (QEST 2008)* (2008).

[2] Fehnker, A., M. Fruth and A. McIver, *Graphical modelling for simulation and formalanalysis of wireless network protocols*, in: *Proc. Workshop on Methods, Models and Tools for Fault-Tolerance (MeMoT'07) at the 7th International Conference on Integrated Formal Methods (IFM'07)*, 2007.

[3] Fehnker, A. and P. Gao, *Formal verification and simulation for performance analysis for probabilistic broadcast protocols*, in: *Ad-Hoc, Mobile, and Wireless Networks, 5th International Conference, ADHOC-NOW 2006, Ottawa, Canada, August 17-19, 2006*, Lecture Notes in Computer Science **4104** (2006), pp. 128–141.

[4] Fehnker, A. and A. McIver, *Formal techniques for the analysis of wireless networks.*, in: T. Margaria, A. Philippou and B. Steffen, editors, *S2nd International Symposium on Leveraging of Formal Maethods, Verification and Validation (IEEE-ISOLA)*, IEEE proceedings, 2006.

[5] Fehnker, A., L. van Hoesel and A. Mader, *Modelling and verification of the lmac protocol for wireless sensor networks*, in: J. Davis and J. Gibbons, editors, *Proceedings of the 6th International Conference on Integrated Formal Methods, IFM 2007, Oxford, Britain*, Lecture Notes in Computer Science (2007), pp. 253–272.

[6] McIver, A., *Quantitative mu-calculus analysis of power management in wireless networks*, in: *International Colloquium of Theoretical Aspects of Computing (ICTAC 2006)*, Lecture Notes in Computer Science (2006).

Probabilistic Analysis of Wireless Systems using Theorem Proving

Osman Hasan[1]

ECE Department
Concordia University
Montreal, Canada

Sofiène Tahar[2]

ECE Department
Concordia University
Montreal, Canada

Abstract

Probabilistic techniques play a major role in the design and analysis of wireless systems as they contain a significant amount of random or unpredictable components. Traditionally, computer simulation techniques are used to perform probabilistic analysis of wireless systems but they provide inaccurate results and usually require enormous amount of CPU time in order to attain reasonable estimates. To overcome these limitations, we propose to use a higher-order-logic theorem prover (HOL) for the analysis of wireless systems. The paper presents a concise description of the formal foundations required to conduct the analysis of a wireless system in a theorem prover, such as, the higher-order-logic modeling of random variables and the verification of their corresponding probabilistic and statistical properties in a theorem prover. In order to illustrate the utilization and effectiveness of the proposed idea for handling real-world wireless system analysis problems, we present an analysis of the automated repeat request (ARQ) mechanism at the logic link control (LLC) layer of the General Packet Radio Service (GPRS), which is a packet oriented mobile data service available to the users of Global System for Mobile Communications (GSM).

Keywords: Formal Methods, GPRS, Higher-Order-Logic, Mechanization of Proofs, Probabilistic Analysis, Theorem Proving, Wireless Networks.

1 Introduction

Wireless communication systems are increasingly being used these days in applications ranging from ubiquitous consumer electronic devices, such as cell phones and computers, to not so commonly used but safety critical domains, such as automated highways and factories, remote tele-medicine and wireless sensor networks. The correctness of operation for these wireless systems is very important due to financial or safety critical nature of their applications. Therefore, quite a significant portion

[1] Email: o_hasan@ece.concordia.ca

[2] Email: tahar@ece.concordia.ca

of the design time of a wireless system is spent on analyzing the designs so that functionality errors can be caught and reliability and performance metrics can be evaluated prior to production. Probabilistic considerations play a significant role in such analysis since wireless systems usually exhibit some random or unpredictable elements. For example, wireless channel parameters are often described in terms of their Probability Mass Functions (PMF) instead of the actual mathematical models for all reflection, diffraction and scattering processes that determine the different multi-path components of a wireless channel. Similarly, probabilistic models are used to describe the mobility of communicating stations. Randomized algorithms and probabilistic analysis are also extensively used in the area of wireless networks. A comprehensive survey in this regard is presented in [41].

Today, simulation is the most commonly used computer based probabilistic analysis technique for wireless systems, e.g., see [39,4,15,25]. Most simulation based wireless system analysis softwares provide a programming environment for defining functions that approximate random variables for probability distributions. The random elements in a given wireless system are modeled by these functions and the system is analyzed using computer simulation techniques [11], such as the Monte Carlo Method [31], where the main idea is to approximately answer a query on a probability distribution by analyzing a large number of samples. Statistical quantities, such as expectation and variance, may then be calculated, based on the data collected during the sampling process, using their mathematical relations in a computer. Due to the inherent nature of simulation coupled with the usage of computer arithmetic, the probabilistic analysis results attained by the simulation approach can never be termed as 100% accurate. Thus, simulation should not be relied upon for the analysis of wireless systems, especially when they are used in safety critical areas, such as, medicine, transportation and military, where inaccuracies in the analysis may even result in the loss of human lives.

In the past couple of decades, formal methods [16] have been successfully used for the precise analysis of a verity of hardware and software systems. The rigorous exercise of developing a mathematical model for the given system and analyzing this model using mathematical reasoning usually increases the chances for catching subtle but critical design errors that are often ignored by traditional techniques like simulation. Given the sophistication of the present age wireless systems and their extensive usage in safety critical applications there is a dire need of using formal methods in this domain. However, due to the random and unpredictable nature of wireless systems, the usage of formal methods has been quite restricted so far. Some major reasons for this include the restriction to handle random behaviors that can be modeled as a Markov chain only and the inability to precisely reason about statistical properties, such as expectation and variance, in the case of state-based approaches and the fear of huge proof efforts involved in reasoning about random components of a wireless system in the case of theorem proving.

We believe that due to the recent developments in the formalization of probability theory concepts in higher-order-logic [24,19,21,18], we are now at the stage where we can handle the analysis of a variety of wireless systems with random components in a higher-order-logic theorem prover [13] with reasonable amount of modeling and verification efforts. The main motivation of using a higher-order-logic theorem prover for this purpose is the ability to formally analyze a broader range of wireless systems by leveraging upon the high expressiveness of the underlying logic.

The foremost requirement for conducting the probabilistic analysis of wireless systems in a higher-order-logic theorem prover is the ability to formalize commonly used random variables in higher-order logic and reason about their corresponding probabilistic and statistical properties in a theorem prover. In this paper, we present a framework illustrating the use of the existing probability theory related higher-order-logic formalizations for fulfilling this requirement and thus in turn analyzing wireless systems. The fact that we are building upon existing formalization tends to minimize the modeling and verification efforts associated with the higher-order-logic theorem proving approach.

In order to illustrate the utilization and effectiveness of theorem proving for handling real-world wireless system analysis problems, we present an analysis of the automated repeat request (ARQ) mechanism at the logic link control (LLC) layer of the General Packet Radio Service (GPRS) standard for Global System for Mobile Communications (GSM) [10]. This analysis is a good representation of a typical wireless system analysis problem that cannot be modeled as a Markov chain and thus cannot be handled by the state based formal analysis approaches. Therefore, the successful handling of this analysis problem clearly indicates the usefulness of the proposed idea. The paper provides a formalization of the ARQ mechanism at the LLC layer of the GPRS standard in higher-order-logic and the formal verification of a couple of probabilistic properties related to the number of LLC frame retransmissions required to successfully transmit a single LLC frame.

The work described in this paper is done using the HOL theorem prover [14], which is based on higher-order logic. The main motivation behind this choice is the fact that most of the work that we build upon is developed in HOL. It is important to note here that the ideas presented in this paper are not specific to the HOL theorem prover and can be adapted to any other higher-order-logic theorem prover as well, such as Isabelle [35], Coq [8] or PVS [36].

The rest of the paper is organized as follows: Section 2 provides a review of the related work. In Section 3, we present a methodology based on existing HOL formalizations of probability theory to the analysis of wireless systems. The ARQ analysis of the LLC layer of the GPRS standard is given in Section 4. Finally, Section 5 concludes the paper.

2 Related Work

Probabilistic model checking [2,38] is the most commonly used formal method in the area of probabilistic analysis of wireless systems. For example, the PRISM model checker [26] has been used to analyze a sub protocol of the IEEE 802.11 standard for wireless local area networks (WLANs) in [27], the IEEE 802.15.4 networking standard in [12] and the Medium Access Control (MAC) protocol SMAC in [3]. Similarly, the ETMCC model checker [23] has been used for the dependability analysis of a variant of the central access protocol of the IEEE 802.11 standard [32]. Just like the traditional model checking, probabilistic model checking involves the construction of a precise state-based mathematical model of the given probabilistic system, which is then subjected to exhaustive analysis to verify if it satisfies a set of formally represented probabilistic properties. Besides the accuracy of the results, the most promising feature of probabilistic model checking is the ability to perform the analysis automatically. On the other hand, it is limited to systems that can only be expressed as probabilistic finite state machines or Markov chains. Another major limitation of the probabilistic model checking approach is state space explosion [7] as has been indicated in [27,12] that increasing the number of communicating stations is not feasible in their analysis due to this problem. Similarly, to the best of our knowledge, it has not been possible to precisely reason about statistical quantities, such as expectation and variance, using probabilistic model checking so far. The most that has been reported in this domain is the approximate evaluation of expected values in a couple of model checkers, such as PRISM [26] and VESTA [40]. For example, in the PRISM model checker, the basic idea is to augment probabilistic models with costs or rewards: real values associated with certain states or transitions of the model. The expectation properties can thus be analyzed in terms of these reward or cost values by PRISM. These expectation properties are expressed and evaluated using computer arithmetic, which introduces some degree of approximation in the results. Similarly, the meaning ascribed to expectation properties is, of course, dependent on the definitions of the costs and rewards themselves and thus there is always some risk of verifying false properties.

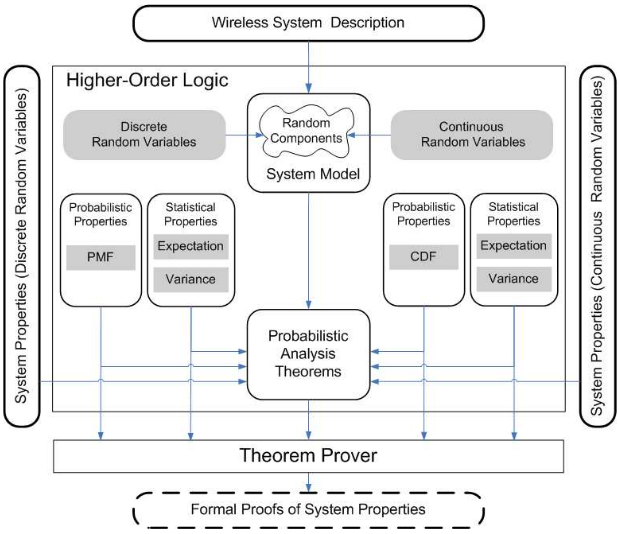
Besides probabilistic model checking, rewriting logic based formal tools have also been used for the probabilistic analysis of wireless systems. For example, Real-Time Maude [33], which is a language and tool supporting the formal specification and analysis of real-time and hybrid systems, has been used for the analysis of the wireless sensor network algorithm OGDC in [34]. But the probabilistic behaviors of the wireless system under analysis are not modeled in formal terms here. Instead, they are analyzed using simulation based methods. Though, formal reasoning about probabilistic specifications is listed as a potential future direction. A possible solution to this aspect would be to explore a combined approach using Real-Time Maude with methods and tools for probabilistic systems, such as PMaude [1].

The proposed higher-order-logic theorem proving based approach tends to overcome the above mentioned limitations of state based formal probabilistic analysis techniques. Due to the high expressibility of higher-order logic, it allows us to analyze a wider range of wireless systems without any modeling limitations, such as the restrictiveness to Markovian models or the state-space explosion problem, and formally verify analytically complex properties, such as expectation and variance. On the other hand, higher-order-logic is an interactive approach and thus requires more human involvement and effort than the state based probabilistic analysis techniques.

To the best of our knowledge, higher-order-logic theorem proving has never been used for the probabilistic analysis of any wireless system so far. Though, some useful research related to the foundations of probabilistic analysis is available in the open literature. The foremost criteria for implementing a theorem proving based probabilistic analysis framework is to be able to formalize and verify random variables in higher-order logic. Hurd's PhD thesis [24] can be considered a pioneering work in this regard as it presents a methodology for the formalization and verification of probabilistic algorithms in the HOL theorem prover. Random variables are basically probabilistic algorithms and thus can be formalized and verified, based on their probability distribution properties, using the methodology proposed in [24]. In fact, [24] presents the formalization of some discrete random variables along with their verification, based on the corresponding PMF properties. Building upon Hurd's formalization framework [24], we have been able to successfully verify the sampling algorithms of a few continuous random variables [19] based on their Cumulative Distribution Function (CDF) properties as well. For comparison purposes, it is frequently desirable to summarize the characteristic of the distribution of a random variable by a single number, such as its expectation or variance, rather than an entire function. For example, it is easier to compare the performance of two wireless communication protocols based on the expected values rather than the CDFs of their message transmission delays. In [21,22], we extended Hurd's formalization framework with a formal definition of expectation. This definition is then utilized to formalize and verify the expectation and variance characteristics associated with discrete random variables that attain values in positive integers only.

3 Probabilistic Analysis Framework

The framework, given in Fig. 1, outlines the main idea behind the theorem proving based probabilistic analysis approach. The shaded boxes in this figure represent the fundamental requirements of conducting probabilistic analysis in a theorem prover. Like all system analysis tools, the input to this framework, depicted by solid rectangles with curved edges, is a description about the wireless system that needs to be analyzed and a set of properties that are required to be checked for the given system. For simplicity, we have divided the system properties into two categories, i.e., system properties related to discrete random variables and system properties related to continuous random variables.



Wireless System Description
Higher-Order Logic
Discrete Random Variables
Random Components
System Model
Continuous Random Variables
Probabilistic Properties
PMF
Statistical Properties
Expectation
Variance
Probabilistic Properties
CDF
Statistical Properties
Expectation
Variance
Probabilistic Analysis Theorems
System Properties (Discrete Random Variables)
System Properties (Continuous Random Variables)
Theorem Prover
Formal Proofs of System Properties

Fig. 1. Theorem Proving based Probabilistic Analysis Framework for Wireless Systems

The first step in conducting probabilistic analysis of a wireless system using a theorem prover is to construct a model of the system in higher-order-logic. For this purpose, the foremost requirement is the availability of infrastructures that allow us to formalize all kinds of discrete and continuous random variables as higher-order-logic functions, which in turn can be used to represent random components of the given wireless system in its higher-order-logic model. The second step in theorem proving based probabilistic analysis is to utilize the formal model of the wireless system to express system properties as higher-order-logic theorems. The prerequisite for this step is the ability to express probabilistic and statistical properties related to both discrete and continuous random variables in higher-order-logic. All probabilistic properties of discrete and continuous random variables can be expressed in terms of their PMFs and CDFs, respectively. Similarly, most of the commonly used statistical properties can be expressed in terms of the expectation and variance characteristics of the corresponding random variable. Thus, we require the formalization of mathematical definitions of PMF, CDF, expectation and variance for both discrete and continuous random variables in order to be able to express the given wireless system's probabilistic and statistical properties as higher-order-logic theorems. The third step for conducting probabilistic analysis in a theorem prover is to formally verify the higher-order-logic theorems developed in the previous step using a theorem prover. For this verification, it would be quite handy to have access to a library of some pre-verified theorems corresponding to some commonly used properties regarding probability distribution functions, expectation and variance. Since, we can build upon such a library of theorems and thus speed up the verification process. Finally the output of the theorem proving based probabilistic analysis framework, depicted by the rectangle with dashed edges, is the formal proofs of system properties that ascertains that the given system properties are valid for the given wireless system.

In order to illustrate the construction details of the framework described above, we now describe the methodologies to fulfill its fundamental requirements.

3.1 Formalization of Discrete Random Variables and Verification of their PMF

A random variable is called discrete if its range, i.e., the set of values that it can attain, is finite or at most countably infinite [43]. Discrete random variables can be completely characterized by their PMFs that returns the probability that a random variable X is exactly equal to some value x, i.e., $Pr(X = x)$.

Discrete random variables are quite frequently used to model random phenomenon in the analysis of wireless systems. For example, the Bernoulli random variable is widely used to model the channel noise behavior [29], the Geometric random variable is often used to model the number of retransmission required to pass a message through a noisy wireless channel [42] and Poisson distribution is typically adopted to model message arrival patterns in wireless network analysis [44].

Discrete random variables can be formalized in higher-order-logic as deterministic functions with access to an infinite Boolean sequence $\mathbb{B}^\infty$; source of an infinite random bits with data type ($num \rightarrow bool$) [24]. These deterministic functions make random choices based on the result of popping the top most bit in the infinite Boolean sequence and may pop as many random bits as they need for their computation. When the functions terminate, they return the result along with the remaining portion of the infinite Boolean sequence to be used by other functions.

Thus, a random variable that takes a parameter of type α and ranges over values of type β can be represented in HOL by the function

$$\mathcal{F} : \alpha \rightarrow B^{\infty} \rightarrow \beta \times B^{\infty}$$

For example, a $Bernoulli(\frac{1}{2})$ random variable that returns 1 or 0 with equal probability $\frac{1}{2}$ can be modeled as follows

```
⊢ bit = λs.  (if shd s then 1 else 0, stl s)
```

where the variable s represents the infinite Boolean sequence and the functions `shd` and `stl` are the sequence equivalents of the list operation *'head'* and *'tail'*. The function `bit` accepts the infinite Boolean sequence and returns a pair with the first element equal to either 0 or 1 and the second element equal to the unused portion of the infinite Boolean sequence, which in this case is the tail of the sequence.

Random variables can also be expressed in a more compact form using the general state-transforming monad where the states are the infinite Boolean sequences.

```
⊢ ∀ a,s.  unit a s = (a,s)
⊢ ∀ f,g,s.  bind f g s = g (fst (f s)) (snd (f s))
```

The HOL functions `fst` and `snd` above return the first and second components of a pair, respectively. The `unit` operator is used to lift values to the monad, and the `bind` is the monadic analogue of function application. All monad laws hold for this definition, and the notation allows us to write functions without explicitly mentioning the sequence that is passed around, e.g., function *bit* can be defined as

```
⊢ bit_monad = bind sdest (λb.  if b then unit 1 else unit 0)
```

where, `sdest` gives the head and tail of a sequence `s` as a pair (*shd* `s`, *stl* `s`).

In order to be able to formally reason about probabilistic properties of random variables, formalized according to the above methodology, we need to formalize a measure space of infinite Boolean sequences. Such a measure space can be used to define a probability function $\mathbb{P}$ from sets of infinite Boolean sequences to *real* numbers between 0 and 1 [24]. Thus, the domain of $\mathbb{P}$ is the set $\mathcal{E}$ of events of the probability. Both $\mathbb{P}$ and $\mathcal{E}$ can be defined using the Carathéodory's Extension theorem, which ensures that $\mathcal{E}$ is a σ-algebra: closed under complements and countable unions. Now, the formalized $\mathbb{P}$ and $\mathcal{E}$ can be used to prove probabilistic properties for random variables such as

$$\vdash \mathbb{P}\ \{\texttt{s}\ |\ \texttt{fst (bit s) = 1}\}\ \texttt{=}\ \frac{1}{2}$$

where $\{x|C(x)\}$ represents a set of all elements x that satisfy the condition C.

The methodology described in this section is quite general and can be utilized to formalize most of the commonly used discrete random variables and formally verify their corresponding PMF relations in a theorem prover. For example, HOL definitions and PMF theorems for the Bernoulli, Uniform, Binomial and Geometric random variables can be found in [24,21,18].

3.2 Formalization of Continuous Random Variables and Verification of their CDF

A random variable is called continuous if it ranges over a continuous set of numbers [43]. A continuous set of numbers, sometimes referred to as an interval, contains all real numbers between two limits. Continuous random variables can be completely characterized by their CDFs that return the probability that a random variable X is exactly less than or equal to some value x, i.e., $Pr(X \leq x)$.

Many wireless system models can only be constructed using continuous random variables. Examples include the modeling of inter-arrival delays between requests to a wireless host using the Exponential random variable [43] and the modeling of mobile nodes displacement by the Uniform random variable [39].

The sampling algorithms for continuous random variables are non-terminating and hence require a different formalization approach than discrete random variables, for which the sampling algorithms are either guaranteed to terminate or satisfy probabilistic termination, meaning that the probability that the algorithm terminates is 1. One approach to address this issue is to utilize the concept of the nonuniform random number generation [11], which is the process of obtaining arbitrary continuous random numbers using a Standard Uniform random number generator. The main advantage of this approach is that we only need to formalize one continuous random variable from scratch, i.e., the Standard Uniform random variable, which can be used to model other continuous random variables by formalizing the corresponding nonuniform random number generation method.

Based on the above approach, [19] presents a methodology, illustrated in Fig. 2, for the formalization of all continuous random variables for which the inverse of the CDF can be represented in a closed mathematical form. The first step in this methodology is the formal specification of the Standard Uniform random variable and the formal verification of this definition by proving the corresponding CDF property. The Standard Uniform random variable can be formalized using the methodology for the formalization of discrete random variables, described in the last section, and the formalization of the mathematical concept of limit of a *real* sequence [17] as the following sampling algorithm

$$\sum_{k=0}^{\infty}(\frac{1}{2})^{k+1}X_k \tag{1}$$

where X_k denotes the outcome of the k^{th} random bit; *True* or *False* represented as 1 or 0 respectively. The formalization and verification details are outlined in [20].

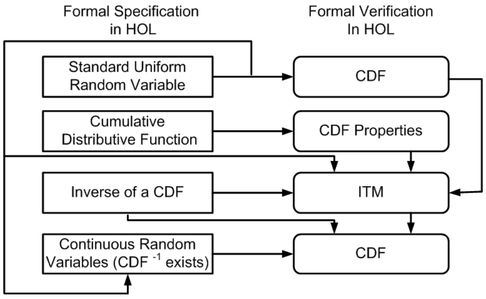

Formal Specification in HOL
Formal Verification In HOL
Standard Uniform Random Variable
Cumulative Distributive Function
Inverse of a CDF
Continuous Random Variables (CDF $^{-1}$ exists)
CDF
CDF Properties
ITM
CDF

Fig. 2. Methodology for the Formalization of Continuous Random Variables

The second step in the methodology for the formalization of continuous probability distributions, given in Fig. 2, is the formalization of the CDF and the verification of its classical properties. This is followed by the formal specification of the mathematical concept of the inverse function of a CDF. This formal specification, along with the formalization of the Standard Uniform random variable and the CDF properties, can be used to formally verify the correctness of the Inverse Transform Method (ITM) [11]. The ITM is a well known nonuniform random generation technique for generating nonuniform random variates for continuous probability distributions for which the inverse of the CDF can be represented in a closed mathematical form. Mathematically, it can be expressed for a random variable X with CDF F using the Standard Uniform random variable U as follows

$$Pr(F^{-1}(U) \leq x) = F(x) \tag{2}$$

and its formal proof can be found [20].

Based on the methodology of Fig. 2, the formalized Standard Uniform random variable can now be used to formally specify any continuous random variable for which the inverse of the CDF can be expressed in a closed mathematical form as $X = F^{-1}(U)$. Whereas, the CDF of this formally specified continuous random variable, X, can be verified, based on simple arithmetic reasoning, using the formal proof of the ITM. Using the above mentioned methodology, [19] presents the formal specification of four commonly used continuous random variables; Exponential, Uniform, Rayleigh and Triangular. The correctness of these random variables is also verified in [19] by proving their corresponding CDF properties in HOL.

3.3 Formalization and Verification of Statistical Properties

The third fundamental component of the wireless system analysis framework, given in Fig. 1, is the ability to formalize and verify statistical properties for random variables. Statistical characteristics, like expectation, play a major role in performance analysis of wireless systems as they tend to summarize the probability distribution characteristics of a random variable in a single number that are easy to compare.

The first and the foremost step towards the ability to reason about statistical properties in a theorem prover is the formalization of an expression for expectation in higher-order logic. Expectation basically provides the average of a random variable, where each of the possible outcomes of this random variable is weighted according to its probability [6]. The expectation for a function of a discrete random variable, which attains values in the positive integers only, is defined as follows [30].

$$Ex\_fn[f(X)] = \sum_{n=0}^{\infty} f(n)Pr(X = n) \tag{3}$$

where X is a discrete random variable and f represents a function of the random variable X. The expression of expectation, given in Equation (3), has been formalized in [22] as a higher-order-logic function using the formalization of the probability function, explained in Section 3.1 of this paper, and the higher-order-logic formalization of the summation of a *real* sequence, given in [17]. The expected value of a discrete random variable that attains values in positive integers can now be defined as a special case of Equation (3) when f is an identity function.

$$Ex[X] = Ex\_fn[(\lambda n.n)(X)] \tag{4}$$

Similarly, a variance function for discrete random variables can be defined in HOL using the expectation definitions given above as follows.

$$Var[X] = Ex\_fn[(\lambda n.(n - Ex[X])^2)X] \tag{5}$$

The above definitions can now be used to formally verify the classical properties of expectation and variance, given in Appendix A. The formal proofs of these properties using the HOL theorem prover can be found in [22,18]. The formal verification of these classical properties not only prove the correctness of the above definitions of expectation and variance but also facilitate the verification of expectation and variance characteristics of discrete random variables in HOL. For illustration purposes, [18] presents the formal verification of the expectation and variance relations for four discrete random variables: Bernoulli, Uniform, Binomial and Geometric.

For formally expressing and verifying statistical characteristics about continuous random variables, we require a higher-order-logic formalization of an integration function that can also handle functions with domains other than real numbers. To the best of our knowledge, a mature formalization for such an integral does not exist in the open literature so far. Thus, reasoning about statistical characteristics regarding continuous random components of a wireless system is not possible as of now. Though, the higher-order-logic formalization of some portions of the Lebesgue integration theory [37] may be extended to tackle such analysis problems.

4 Analysis of ARQ at the LLC Layer of GPRS

Due to the rapid development in mobile computing devices and emerging market of multimedia communications, the data-bearer service standard GPRS [5], which operates in packet-switched mode, was introduced as part of GSM phase 2+. GPRS uses the existing GSM infrastructure to provide high-speed (up to 270 kb/s) data communications, which is ideal for Multimedia Messaging Service (MMS) and for Internet communication services such as email and World Wide Web access. The biggest challenge in the design of GPRS is to maintain reliable data transfers without incurring too much delays under the erroneous nature of a wireless channel (due to distance losses, shadowing, and multipath fading). Higher layer protocols, such as the Transmission Control Protocol (TCP), are usually designed for wired channels that exhibit very low error rates and thus perform poorly if they are made responsible for the reliability of data transfers using a wireless channel [28]. Hence, the lower layers in GPRS stacks, e.g., the LLC and radio link control/medium access control (RLC/MAC), must be designed to address these issues of high error rates and higher layer-performance concerns.

The GPRS data transfer reliability model is as follows. A stop-and-wait ARQ [29] mechanism is implemented at the LLC to retransmit the erroneous LLC frames in order to ensure reliable transfers at the LLC peer-to-peer link. The LLC frames are passed to the RLC/MAC layer first, where they are segmented into RLC/MAC blocks of fixed size. These blocks are then transmitted through the radio channel one by one. The RLC/MAC layer also provides an ARQ mechanism and thus provides further error recovery over the radio channel. Our focus in this paper is to formally prove, using the theorem proving based probabilistic analysis approach described in Section 3, a probabilistic relation for the number of LLC frame retransmissions required for successfully transmitting a single LLC frame in terms of the probability of successful transmission of a single RLC/MAC block, say p, through the radio channel and the LLC frame size in RLC/MAC blocks, say n. Such an expression plays a vital role in estimating the LLC frame size of the GPRS, for a given channel, to maximize performance. We also formally verify that the GPRS data reliability model ensures successful transmission of every LLC frame with probability 1. Our analysis approach is mainly inspired by a paper-and-pencil based analytical analysis of a similar problem presented in [10].

The first step according to the probabilistic analysis framework, given in Fig 1, is to describe the given system as a higher-order-logic function while representing its random components as random variables. In case of the above mentioned GPRS analysis problem, we need to develop a higher-order-logic function that describes the LLC frame transmission behavior in terms of the parameters p and n. The random component in this system is the behavior of the wireless channel, which allows data blocks to pass through with probability p. We formalized the LLC frame transmission behavior as a higher-order-logic predicate, i.e., a function that returns a Boolean value. Our predicate accepts three parameters: n, p and k, where k represents the number of transmission attempts. It returns *True* if all n RLC/MAC blocks are successfully transmitted within k attempts and *False* otherwise. The predicate can be expressed recursively in HOL as follows

```
Definition 1: LLC Frame Transmission Behavior
    ⊢ ∀ k p.  (llc_trans 0 k p = unit (True)) ∧
      ∀ n k p.  (llc_trans (n + 1) k p =
        bind (llc_trans n k p) (λa.  bind (prob_bino k p)
          (λb.  unit (if (b = 0) then False else a))))
```

where the function `prob_bino` represents the formalized Binomial random variable, given in [18]. A Binomial(k, p) random variable models an experiment that counts the number of successes in k independent Bernoulli(p) trials [9]. Thus, it is used in the above predicate to estimate the number of successful transmissions in k transmission attempts of an RLC/MAC block, as the behavior of a noisy wireless channel with successful transmission probability p can be modeled by a Bernoulli(p) random variable. The predicate `llc_trans` recursively checks the number of successful transmissions for each one of the n RLC/MAC blocks and returns *False* if one or more of these blocks have no successful transmission in the k transmission attempts.

The second step according to the framework, given in Fig 1, is to utilize the formal model of the system to express the properties of interest as higher-order-logic theorems. In our case, we are interested in the probability that a single LLC frame consisting of n blocks is transmitted within k transmission attempts. The HOL theorem corresponding to this property can be expressed based on the predicate `llc_trans`, given in Definition 1, as follows

Theorem 1:
$$\vdash \forall\ n\ k\ p\ k.\ \ (0 \leq p) \wedge (p \leq 1) \Rightarrow$$
$$(\mathbb{P}\ \{s\ |\ (\texttt{fst}\ (\texttt{llc\_trans}\ n\ k\ p\ s))\} = (1 - (1-p)^{k})^{n})$$

where $\mathbb{P}$ represents the formalization of the probability function, explained in Section 3. We verified the above theorem in HOL and the proof is primarily based on the PMF theorem of the Binomial random variable, verified in [18], and some arithmetic and probabilistic reasoning. For illustration purposes the proof sketch is provided in Appendix B.

Next, we formally prove the correctness of the GPRS data reliability model by verifying that the probability of successfully transmitting an LLC frame approaches 1 as the number of transmission trials becomes very very large. This property can be expressed in HOL, based on the predicate `llc_trans`, as follows

`Theorem 2:`

$$\vdash \forall\ \texttt{n k p k}.\ \ (\texttt{0} \leq \texttt{p}) \wedge (\texttt{p} \leq \texttt{1}) \Rightarrow$$
$$\texttt{lim}_{\texttt{k}\to\infty}\ (\mathbb{P}\ \{\texttt{s} \mid (\texttt{fst (llc\_trans n k p s)})\}) = \texttt{1}$$

using the HOL formalization of limit of a *real* sequence, given in [17]. The HOL proof of Theorem 2 is based on Theorem 1 and some classical properties of limit of a real sequence, verified in [17].

The above example clearly demonstrates the effectiveness of the theorem proving based wireless system analysis approach. Due to the formal nature of the model and inherent soundness of theorem proving, we have been able to verify probabilistic properties of the given system with 100% precision; a novelty which is not available in simulation. Similarly, due to the high expressibility of higher-order logic we have been able to formally reason about a problem that cannot be described as a Markov chain and thus cannot be analyzed using a probabilistic model checker. These additional benefits come at the cost of the time and effort spent, while formalizing the system and formally reasoning about its properties, by the user. But, the fact that we were building on top of already verified results in the theorem prover helped significantly in this regard as the analysis, described in this section, only consumed approximately 40 man-hours by an expert HOL user.

5 Conclusions

This paper advocates the usage of higher-order-logic theorem proving for the probabilistic analysis of wireless systems in order to be able to precisely analyze a wide range of problems. This approach can thus be of great benefit for the analysis of wireless systems used in safety critical applications, such as medicine and transportation. The paper provides a theorem proving based generic methodology for the probabilistic analysis of wireless systems. For illustration purposes, we present an analysis of ARQ mechanism at the LLC layer of the GPRS. To the best of our knowledge, this is the first time that a theorem prover has been used to conduct the probabilistic analysis of a wireless system.

There are many research directions in the field of using theorem provers for the probabilistic analysis of wireless systems that need to be explored. A couple of interesting ones include the ability to formalize and reason about statistical properties about continuous random variables and the ability to model Markov chains in higher-order-logic and reason about their probabilistic and statistical properties in a higher-order-logic theorem prover.

References

[1] G. Agha, J. Meseguer, and K. Sen. PMaude: Rewrite-based Specification Language for Probabilistic Object Systems. *Electronic Notes in Theoretical Computer Science*, 153(2):213–239, 2006.

[2] C. Baier, B. Haverkort, H. Hermanns, and J.P. Katoen. Model Checking Algorithms for Continuous time Markov Chains. *IEEE Transactions on Software Engineering*, 29(4):524–541, 2003.

[3] P. Ballarini and A. Miller. Model Checking Medium Access Control for Sensor Networks. In *International Symposium on Leveraging Applications of Formal Methods, Verification and Validation*, pages 255–262. IEEE, 2006.

[4] P. Barker, V. Vitsas, and A.C. Boucouvalas. Simulation Analysis of Advanced Infrared (alr) MAC Wireless Communications Protocol. *Circuits, Devices and Systems*, 149(3):193–197, 2002.

[5] C. Bettstetter, H. Vögel, and J. Eberspächer. GSM Phase 2+ General Packet Radio Service GPRS: Architecture, Protocols, and Air Interface. *IEEE Communication Surveys*, 2(3), 1999.

[6] P. Billingsley. *Probability and Measure.* John Wiley, 1995.

[7] E.M. Clarke, O. Grumberg, and D.A. Peled. *Model Checking.* The MIT Press, 2000.

[8] COQ. http://pauillac.inria.fr/coq/, 2008.

[9] M. DeGroot. *Probability and Statistics.* Addison-Wesley, 1989.

[10] C. Demetrescu. LLC-MAC Analysis of General Packet Radio Service in GSM. *Bell Labs Technical Journal*, 4(3):37–50, 1999.

[11] L. Devroye. *Non-Uniform Random Variate Generation.* Springer-Verlag, 1986.

[12] M. Fruth. Probabilistic Model Checking of Contention Resolution in the IEEE 802.15.4 Low-Rate Wireless Personal Area Network Protocol. In *International Symposium on Leveraging Applications of Formal Methods, Verification and Validation*, pages 290–297. IEEE, 2006.

[13] M.J.C. Gordon. Mechanizing Programming Logics in Higher-0rder Logic. In *Current Trends in Hardware Verification and Automated Theorem Proving*, pages 387–439. Springer, 1989.

[14] M.J.C. Gordon and T.F. Melham. *Introduction to HOL: A Theorem Proving Environment for Higher-Order Logic.* Cambridge University Press, 1993.

[15] R. Gowaikar, B. Hochwald, and B. Hassibi. Communication over a Wireless Network with Random Connections. *IEEE Transactions on Information Theory*, 52(7):2857–2871, July 2006.

[16] A. Gupta. Formal Hardware Verification Methods: A Survey. *Formal Methods in System Design*, 1(2-3):151–238, 1992.

[17] J. Harrison. *Theorem Proving with the Real Numbers.* Springer, 1998.

[18] O. Hasan and S. Tahar. Formal Verification of Tail Distribution Bounds in the HOL Theorem Prover. *Mathematical Methods in the Applied Sciences.* In-print.

[19] O. Hasan and S. Tahar. Formalization of the Continuous Probability Distributions. In *Automated Deduction*, volume 4603 of *LNAI*, pages 3–18. Springer, 2007.

[20] O. Hasan and S. Tahar. Formalization of the Standard Uniform Random Variable. *Theoretical Computer Science*, 382(1):71–83, 2007.

[21] O. Hasan and S. Tahar. Verification of Expectation Properties for Discrete Random Variables in HOL. In *Theorem Proving in Higher-Order Logics*, volume 4732 of *LNCS*, pages 119–134. Springer, 2007.

[22] O. Hasan and S. Tahar. Formal Verification of Expectation and Variance for Discrete Random Variables. Technical Report, Concordia University, Montreal, Canada, June 2007; http://hvg.ece.concordia.ca/Publications/TECH_REP/FVEVDR_TR07.

[23] H. Hermanns, J-P. Katoen, J. Meyer-Kayser, and M. Siegle. A Tool for Model-Checking Markov Chains. *Software Tools for Tech. Transfer*, 4(2):153–172, 2003.

[24] J. Hurd. *Formal Verification of Probabilistic Algorithms.* PhD Thesis, University of Cambridge, Cambridge, UK, 2002.

[25] A. Jain, S.S. Pawar, R. Upadhyay, and S.V. Charhate. Monte Carlo Simulation Based Error Performance Analysis of DS-CDMA System. In *Asia International Conference on Modelling and Simulation*, pages 230–233. IEEE Computer Society, 2008.

[26] M. Kwiatkowska, G. Norman, and D. Parker. Quantitative Analysis with the Probabilistic Model Checker PRISM. *Electronic Notes in Theoretical Computer Science*, 153(2):5–31, 2005. Elsevier.

[27] M(.) Kwiatkowska, G. Norman, and J. Sproston. Probabilistic Model Checking of the IEEE 802.11 Wireless Local Area Network Protocol. In *Joint International Workshop on Process Algebra and Probabilistic Methods, Performance Modeling and Verification*, volume 2399 of *LNCS*, pages 169–187. Springer, 2002.

[28] T.V. Lakshman and U. Madhow. The Performance of TCP/IP for Networks with High Bandwidth-delay Products and Random Loss. *IEEE/ACM Transactons on Networking*, 5(3):336–350, 1997.

[29] A. Leon Garcia and I. Widjaja. *Communication Networks: Fundamental Concepts and Key Architectures.* McGraw-Hill, 2004.

[30] A. Levine. *Theory of Probability.* Addison-Wesley series in Behavioral Science, Quantitative Methods, 1971.

[31] D.J.C. MacKay. Introduction to Monte Carlo Methods. In *Learning in Graphical Models, NATO Science Series*, pages 175–204. Kluwer Academic Press, 1998.

[32] M. Massink, D. Latella, and J-P. Katoen. Model Checking Dependability Attributes of Wireless Group Communication. In *Dependable Systems and Networks*, pages 711–720. IEEE, 2004.

[33] P.C. Olveczky and J. Meseguer1. Specification and Analysis of Real-Time Systems Using Real-Time Maude. In *Fundamental Approaches to Software Engineering*, volume 2984 of *LNCS*, pages 354–358. Springer, 2004.

[34] P.C. Ölveczky and S. Thorvaldsen. Formal Modeling and Analysis of the OGDC Wireless Sensor Network Algorithm in Real-Time Maude. In *Formal Methods for Open Object-based Distributed Systems*, volume 4468 of *LNCS*, pages 122–140, 2007.

[35] L.C. Paulson. *Isabelle: A Generic Theroem Prover*, volume 828 of *LNCS*. Springer, 1994.

[36] PVS. http://pvs.csl.sri.com, 2008.

[37] S. Richter. *Formalizing Integration Theory, with an Application to Probabilistic Algorithms.* Diploma Thesis, Technische Universitat Munchen, Department of Informatics, Germany, 2003.

[38] J. Rutten, M. Kwaiatkowska, G. Normal, and D. Parker. *Mathematical Techniques for Analyzing Concurrent and Probabilisitc Systems*, volume 23 of *CRM Monograph Series.* American Mathematical Society, 2004.

[39] P. Santi, D.M. Blough, and F. Vainstein. A Probabilistic Analysis for the Range Assignment Problem in Ad Hoc Networks. In *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 212–220. ACM, 2001.

[40] K. Sen, M. Viswanathan, and G. Agha. VESTA: A Statistical Model-Checker and Analyzer for Probabilistic Systems. In *Proc. IEEE International Conference on the Quantitative Evaluation of Systems*, pages 251–252, 2005.

[41] A. Srinivasan. Randomized Algorithms and Probabilistic Analysis in Wireless Networking. In *Stochastic Algorithms, Foundations, and Applications*, volume 4665 of *LNCS*, pages 54–57. Springer, 2007.

[42] P. Tran-Gia and K. Leibnitz. Teletraffic Models and Planning in Wireless IP Networks. In *Wireless Communications and Networking Conference*, volume 2, pages 598–602. IEEE, 1999.

[43] R.D. Yates and D.J. Goodman. *Probability and Stochastic Processes: A Friendly Introduction for Electrical and Computer Engineers.* Wiley, 2005.

[44] J. Zheng and M. J. Lee. *A Comprehensive Performance Study of IEEE 802.15.4*. IEEE Press, 2004.

Appendix A: Expectation and Variance Properties

No.	Property	Mathematical Representation
1	Linearity of Expectation 1	$Ex[\sum_{i=1}^{n} R_i] = \sum_{i=1}^{n} Ex[R_i]$
2	Linearity of Expectation 2	$Ex[a + bR] = a + bEx[R]$
3	Markov's Inequality	$Pr(X \geq a) \leq \frac{Ex[X]}{a}$
4	Alternate Definition of Variance	$Var[R] = Ex[R^2] - (Ex[R])^2$
5	Linearity of Variance	$Var[\sum_{i=1}^{n} R_i] = \sum_{i=1}^{n} Var[R_i]$
6	Chebyshev's Inequality	$Pr(\lvert X - Ex[X]\rvert \geq a) \leq \frac{Var[X]}{a^2}$

Table 1
Expectation and Variance Properties

Appendix B: HOL Proof for Theorem 1

We proceed to verify Theorem 1 by performing inductance on the variable n, which generates the following two subgoals.

$$\mathbb{P}\{\texttt{s} \mid \texttt{fst (llc\_trans 0 k p s) }\} \;=\; (\texttt{1} \;-\; (\texttt{1} \;-\; \texttt{p})^{\texttt{k}})^{\texttt{0}}$$

$$\begin{gathered}\mathbb{P}\{\texttt{s} \mid \texttt{fst (llc\_trans n k p s)}\} = \;(\texttt{1} \;-\; (\texttt{1} \;-\; \texttt{p})^{\texttt{k}})^{\texttt{n}} \;\Rightarrow \\ \mathbb{P}\{\texttt{s} \mid \texttt{fst (llc\_trans (n + 1) k p s) }\} \;=\; (\texttt{1} \;-\; (\texttt{1} \;-\; \texttt{p})^{\texttt{k}})^{(\texttt{n + 1})}\end{gathered}$$

The base case, i.e., the first subgoal above, can be simply proved using the definition of the function `llc_trans`, given in Definition 1, the probability law $P(\bigcup) = 1$ and some arithmetic reasoning. Whereas, we proceed with the proof of the step case, i.e., the second subgoal above, by rewriting it using the definition of the function `llc_trans` and simplifying it using some arithmetic reasoning as follows.

$$\begin{gathered}\mathbb{P}\{\texttt{s} \mid \texttt{fst (llc\_trans n k p s)}\} = \;(\texttt{1} \;-\; (\texttt{1} \;-\; \texttt{p})^{\texttt{k}})^{\texttt{n}} \;\Rightarrow \\ \mathbb{P}\{\texttt{s} \mid \texttt{(fst (llc\_trans n k p s)} \wedge \\ \neg\texttt{(fst (prob\_bino k p (snd (llc\_trans n k p s))) = 0)}\} \\ = \;(\texttt{1} \;-\; (\texttt{1} \;-\; \texttt{p})^{\texttt{k}})^{\texttt{n}}(\texttt{1} \;-\; (\texttt{1} \;-\; \texttt{p})^{\texttt{k}})\end{gathered}$$

Now, using the statistical independence between the two events in the set on the LHS of the conclusion of the above implication and the probability law $P(A \cap B) = P(A)P(B)$, the above subgoal can be simplified as follows.

$$\begin{gathered}\mathbb{P}\{\texttt{s} \mid \texttt{fst (llc\_trans n k p s)}\} = \;(\texttt{1} \;-\; (\texttt{1} \;-\; \texttt{p})^{\texttt{k}})^{\texttt{n}} \;\Rightarrow \\ \mathbb{P}\{\texttt{s} \mid \texttt{(fst (llc\_trans n k p s)}\} \\ \mathbb{P}\{\texttt{s} \mid \neg\texttt{(fst (prob\_bino k p (snd (llc\_trans n k p s))) = 0)}\} \\ = \;(\texttt{1} \;-\; (\texttt{1} \;-\; \texttt{p})^{\texttt{k}})^{\texttt{n}}(\texttt{1} \;-\; (\texttt{1} \;-\; \texttt{p})^{\texttt{k}})\end{gathered}$$

Using the assumption in the above subgoal along with some arithmetic reasoning we get the following subgoal

$$\mathbb{P}\{\texttt{s} \mid \neg(\texttt{fst (prob\_bino k p (snd (llc\_trans n k p s))) = 0})\} \\ = \texttt{(1 - (1 - p)}^{\texttt{k}}\texttt{)}$$

which can be rewritten using the complement law of the probability $P(\bar{A}) = 1 - P(A)$ as follows

$$\mathbb{P}\{\texttt{s} \mid \texttt{(fst (prob\_bino k p (snd (llc\_trans n k p s))) = 0)}\} = \texttt{(1 - p)}^{\texttt{k}}\texttt{)}$$

This subgoal can now be verified using the PMF relation of the Binomial random variable (`prob_bino`), given in [18], along with some arithmetic reasoning. This also concludes the proof of Theorem 1 in HOL.

Performance Analysis of IEEE 802.11b Wireless Networks with Object Oriented Petri Nets

Aladdin Masri[1] Thomas Bourdeaud'huy[2] Armand Toguyeni[3]

LAGIS - CNRS UMR 8146
Ecole Centrale de Lille - Cit Scientifique BP 48
59651 Villeneuve d'ASCQ, France

Abstract

Communication protocols are often investigated using simulation. This paper presents a performance study of the distributed coordination function of 802.11 networks. Firstly, our study illustrates the different classes of Petri Nets used for modeling network protocols and their robustness in modeling based on formal methods. Next we propose a detailed 802.11b model based on Object-oriented Petri Nets that precises backoff procedure and time synchronization. Then, performance analyses are evaluated by simulation for a dense wireless network and compared with other measurements approaches. Our main goal is to propose a modular model that will enable to evaluate the impact of network performances on the performances of distributed discrete event systems.

Keywords: LAN protocols modeling, Petri Nets, Performance Analysis, 802.11b Standard, Simulation.

1 Introduction

Wireless technology has become popular to access to the internet and communication networks. The IEEE 802.11 offers the possibility to assign part of the radio channel bandwidth to be used in wireless networks. IEEE 802.11 has two ways to access the channel: *Point Coordination Function PCF* and *Distributed Coordination Function DCF* that uses CSMA/CA which allows sharing the channel fairly based on best effort. The characteristic of wireless networks vary from the wired networks. The method to access the channel, in DCF mode, requires checking if the channel is idle for more than a *DIFS* (Distributed Inter Frame Space). Then, it begins its transmission after a random backoff based on the value of the contention window *CW*. It must receive an acknowledgment from the destination, after a *SIFS* (Short IFS) time, to guarantee a successful transmission, otherwise it will assume that the frame is in collision and retransmit it as above.

[1] Email: aladdin.masri@ec-lille.fr

[2] Email: thomas.bourdeaud_huy@ec-lille.fr

[3] Email: armand.toguyeni@ec-lille.fr

In this paper we propose an Object-Oriented Petri Nets modeling approach that is a brick to model the impact of networks' protocols on the performances of distributed discrete event systems. We develop a model that fulfills all the constraints of communication protocols. The main constraints are timing and synchronization of workstations especially for distributed systems. We also take the stochastic requirement into consideration for the bit rate errors and for the transmission depending on the services. Another constraint is the ability to analyze the impact of others traffics on a specific one between two workstations. The approach also proposes in addition the modeling of backoff, collision procedure and a dynamic length of data frames.

The paper is organized as follows. Section 2 gives a mathematical definition and a comparison of the different classes of Petri Nets. We discuss the benefits and weakness points for each class in the modeling of communication protocols. Section 3 gives a brief introduction to IEEE 802.11b DCF and presents our model. At the end, performance analysis is validated by means of simulation.

2 Petri Nets For Modeling Network Protocols

Petri nets have been proposed by C. A. Petri in 1962 [1]. Petri nets are a powerful modeling formalism in computer science, system engineering and many other disciplines. They are used to study and describe different types of systems: distributed, parallel, and stochastic; mainly *discrete event systems.* Petri nets are in two forms: *mathematical and graphical.*

2.1 Modeling with Ordinary Petri Nets

An ordinary Petri net $\boldsymbol{N=(P,\ T,\ A,\ m_0)}$ can be defined as a bipartite directed graph, where:

- **P** and **T** are the sets of nodes respectively called places and transitions ($|P| = m, |T| = n$);
- **A: P×T ∪ T×P → N** is the weighted flow relation representing the arcs;
- **m$_0$: P → N** is a mapping associating to each place p∈**P**, an integer $m_0(p)$ called the initial marking of the place p.

The marking of a Petri net can be modified by the firing of transitions. A transition t is fireable from a marking m_a (denoted by $m_a[t\rangle$), when $\forall$p$\in{}^o\mathbf{t}$ with $^o\mathbf{t}$={p∈**P** such as **A**(p, t)>0}, $m_a(p) \geq \boldsymbol{A}(p,\ t)$. If this condition is satisfied, a new marking m$_k$ is produced from the marking m$_a$(denoted by $m_a[t\rangle m_k$): $\forall$p∈**P**, m$_k$(p)= m$_a$(p) + **A**(p, t) + **A**(t, p).

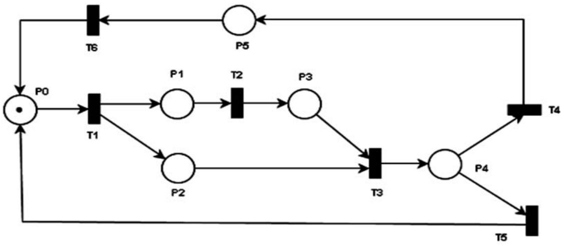


Fig. 2.1 Example of an Ordinary Petri Net

Fig. 2.1 shows an example of an ordinary Petri net. In this figure one can see some characteristics of a Petri net. The transition T2 cannot be fired before the firing of T1. This characteristic is called the *sequential execution.* T3 is a *synchronisation transition* since it is enabled as soon as P2 and P3 have tokens. Transition T4 and T5 are in *conflict* since only one of them can be fired when P4 receives a token. However, there are some problems to model computer protocols with ordinary Petri nets:

(i) *Time modeling.* Ordinary Petri nets do not handle time. This makes it difficult or even impossible to model communication protocols with such Petri nets because time is one of the main features of network protocols.

(ii) *Priority and stochastic modeling.* These characteristics do not exist in ordinary Petri nets. This does not solve the conflict problem or cannot define a probability to fire such transitions.

2.2 Modeling with Timed Petri Nets

Timed Petri nets are a class of Petri nets. It was introduced by C. Ramchandani in 1974 [2]. It is seen as $\boldsymbol{N= (P,\ T,\ A,\ m_0,\ \tau)}$ where $\boldsymbol{(P,\ T,\ A,\ m_0)}$ is an ordinary Petri net, and τ***:*** $\boldsymbol{T \rightarrow R^+}$ is a function that associates time delays to transitions.

In a timed Petri net, it is not necessary that a clock is associated to every transition. However, the time delays associated to the transitions modify the *marking validity conditions.* When a transition is fired, the token(s) in the input place(s) seems as it *disappears* and then it *reappears* after a period equals to delay associated to that transition. As a result, the beginning and end moments of transitions firing play a fundamental rule in the behavior of the timed Petri net which means one must take care of the delays desired to fire transitions.

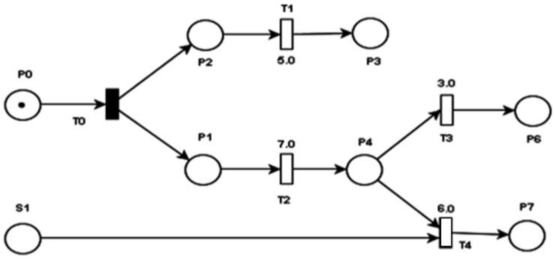


Fig 2.2 A Timed Petri Net

One can see in Fig. 2.2 that T0 is an immediate transition while T1, T2, T3 and T4 are timed transitions. So, after firing T0, both T1 and T2 are enabled. T1 can be fired after 5 units of time, and T2 can be fired after 7 units of time. T3 is now enabled but it has to wait 3 units of time before firing. Suppose that S1 (another workstation sending data) had a token before, T4 is now enabled. The choice of T3 or T4 determines to which place the token in P4 will go and after what time.

This may help in solving the conflict problem since both transitions are controlled by time, but still not for immediate transitions. However, this does not solve the stochastic problem. Suppose 90% of the packets of P4 may go to P6 and 10% may go to P7 (the bit rate error modeling in communication protocols), the timed Petri nets does not answer this characteristics. Another problem that one cannot model with this class is the time variation or intervals. As the time delays associated to transitions is constant, the occurrence possibility in an interval of time cannot be modeled. As an example, the length of a packet sent on a network varies from ˜ 60 to 1514 bytes. The time needed to send such packets depends on the length of that packet. With constant timing values, this action cannot be modeled easily or one must complexify the model.

2.3 Modeling with Time Petri Nets

Time Petri nets [3] [4] is a more powerful formalism used to model systems where time is the main constraint such as communication protocols and real-time systems. A TPN is a five-tuple $\boldsymbol{N=(P,\ T,\ A,\ m_0,\ \tau_s)}$ where τ_s**:** $\boldsymbol{T} \rightarrow \boldsymbol{R}^+ \times \boldsymbol{R}^+ \cup \{+\infty\}$ is a function called *Static Interval function.* Time is represented in intervals with lower min and upper max limits which make it easy to model events with unknown occurring time. The two limits min and max (with $0 \leq$ min $\leq$ max, min$\in\mathbf{R}^+$ and max$\in\mathbf{R}^+\cup\{+\infty\}$) are associated to each transition. These limits are related to the date when $\mathbf{t}_i$ was enabled for the last time. Let θ be the date when $\mathbf{t}_i$ becomes enabled; then $\mathbf{t}_i$ cannot be fired before θ*+min* and must fire no later than θ*+max* (if max is finite), except if the fire of another transition $\mathbf{t}_j$ un-enables $\mathbf{t}_i$ before it is fired. Transition firings have no durations.

The transition firing in a Time Petri Net has two firing semantics. The first semantics is called the *strong firing semantics*, which impose that any enabled transition must be fired at its latest firing time at most. On the contrary, when using the *weak firing semantics*, the firing time of a transition is not constrained by firing conditions over other transitions. In this paper, we will use the strong firing semantics for the watchdog needs. In Fig. 2.3, in strong firing semantics, the transition T1 cannot be fired after 9 units of time since T0 must be fired before 9 units of time. T2 is an immediate transition.

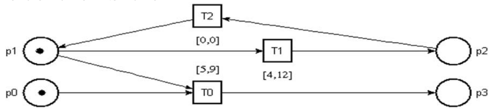


Fig. 2.3 A Time Petri Net

In some new tools, the TPN was improved with priority selection. In Fig. 2.4, transition T1 is fireable between 3 and 11 units of time. However, T0 has priority over T1, so T1 can be fired between 3 and 5 units of time but not later since T0 has priority at that time.

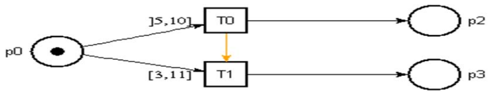


Fig. 2.4 Priority in TPN

Problems are not still all solved. The priority may be now solved but what about modeling complexity, percentage distribution or data addressing. In wireless networks, a workstation trying to send a packet must wait for a backoff delay before sending its packet. The backoff value is a random value between 0 and CW multiplied by the slot time. Fig. 2.5 shows how to model such action. This is just for random number, but what about data addressing or percentage distribution? TPN does not answer these questions now since it has no token identification or probability functions. In addition, if one insists, the complexity of the model prevents any analysis or what is known as the combinatorial explosion problem. For each workstation, the value of CW is at first 16, but after each collision (no acknowledgement received) the current CW value is multiplied by two, until it reaches 1024. So if one tries to get the state classes, it would be impossible since one have this huge number of tokens in just one place.

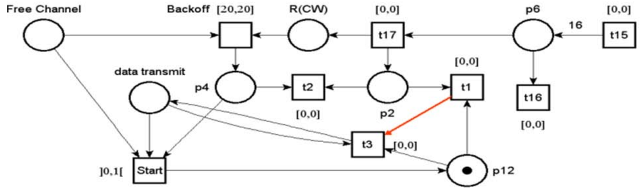


Fig 2.5 Random Backoff modeled with TPN

2.4 Modeling with Stochastic Petri Net

Stochastic Petri Nets [5] were proposed to integrate formal description, proof of correctness, and performance evaluation. They are Petri nets in which stochastic firing times are associated with transitions.

A Stochastic Petri Net is a tuple $\boldsymbol{N=(P,\ T,\ A,\ m_0,\ \Gamma)}$ where Γ***: T→pdf*** is a set of firing rates, and *pdf* is the *probability density function*. The entry $\delta_i \in \Gamma$ is an exponential distributed random variable, whose pdf is a negative exponential, associated with transition $\mathbf{t}_i$. The firing rate of any transition $\mathbf{t}_i$ may be marking-dependent, so it is necessary to be written as $\delta_i(\mathrm{M}_j)$. Thus, the average firing delay of transition $\mathbf{t}_i$ in marking M_j is $[\delta_i(\mathrm{M}_j)]^{-1}$. Since the rate is marking-dependent, when entering a marking, the transition with the minimum firing delay will be fired. Knowing that all the firing delays have *exponential pdf*, this allows saying that the probability for a given transition $\mathbf{t}_i$ with the minimum delay as:

$$\mathrm{P}(\mathrm{t}_i,\mathrm{M}_j)=\frac{\delta_i(M_j)}{\sum_{k:t_k \in X(M_j)} \delta_k(M_j)}$$

Where $\mathrm{X}(\mathrm{M}_j)$ is the set of all enabled transitions in the marking M_j.

So suppose there are three enabled transitions with firing rate $\alpha 1, \alpha 2, \alpha 3$ with minimum delay for $\mathbf{t}_1$, and then the probability of firing $\mathbf{t}_1$ is:

$$\mathrm{P}(\mathrm{t}_1)=\frac{\alpha 1}{\alpha 1+\alpha 2+\alpha 3}$$

The Generalized SPN, Fig. 2.6, is a subclass of the stochastic Petri nets, which allows immediate transitions in the net (which is not the case for SPN). A priority zero is given to timed transition while the immediate transitions own a priority higher or equal to 1.

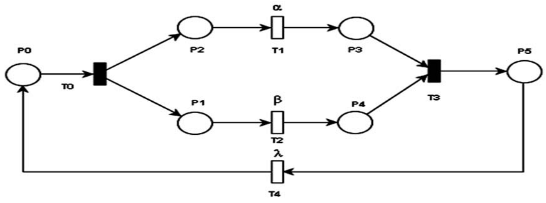


Fig. 2.6 A Generalized SPN

Stochastic Petri nets can now answer some problems and find solutions to them. But still not all the modeling problems are solved yet. Consider the communication between different workstations on the net, a workstation trying to communicate with another workstation must give the destination address so that the other workstations either forward the request or if it is the destination it will pick it up and stop forwarding the packet. In nearly all situations, the destination workstation sends an acknowledgement to the source workstation informing the reception of the message, otherwise it will repeat the transmission, as the wireless protocols [6] for example.

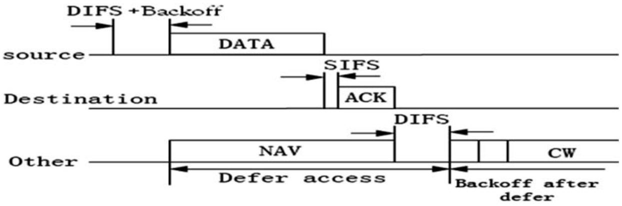


Fig. 2.7 Wireless Protocol message exchange process

This process needs to "label" the token with the name of the source, destination and the message an in Fig. 2.7. The stochastic Petri nets do not have the capacity to do this since tokens are all of the same type and have no modifiers.

2.5 Modeling with Colored Petri Nets

Colored Petri nets [7] have different characteristics from other classes, where token(s) and places are attached with a color identifying the type of that token and place. A CPN is a tuple $\boldsymbol{N=(P, T, A, m_0, \Sigma, \Lambda, G, E, I)}$ where:

- Σ is a finite set of non-empty color sets.
- Λ is a color function, Λ: P $\rightarrow \Sigma$.
- **G** is a guard function, G: T $\rightarrow$ Boolean expression, where: $\forall$t$\in$T: [Type(G(t)) = $B_{exp} \wedge$ Type(Var(G(t))) $\subseteq \Sigma$]
- **E** is an arc expression function, E: A $\rightarrow$ E(a), where: $\forall$a$\in$A: [Type(E(a))= Λ(p(a))$\wedge$Type(Var(E(a)))$\subseteq\Sigma$]], p(a) is the place of arc **a**.
- **I** is an initialization function, I: P$\rightarrow$**a** closed expression I(p) (without variables) where: $\forall$p$\in$P: [Type(I(p)) = Λ(p)]

From the above definition one can say that the color function defines the type (called multi-set type) of values in each place. Arcs' inscriptions must be a non-empty expression type that matches the color of the place to where it is connected (to fire a transition). The initial marking m_0 is obtained by evaluating the initialization expressions: $\forall$p$\in$P: m_0(p)= I(p) where m_0(p)$\in\Lambda$(p).

The firing of a transition in a CPN must satisfy some conditions:

(i) Input places of a transition $\mathbf{t}_i$ must contain the number of tokens enabling that transition: $\forall$p$\in{}^o\mathbf{t}$ with ${}^o\mathbf{t}$= $\forall$p$\in$**P** such as **A**(p,t)>0, m_j(p) $\geq$ **A**(p, t) and Type(E(a))= Λ(p(a))

(ii) The guard function associated with that transition must be true to enable the transition: G(t_i) = True

(iii) The output tokens (tokens in output places) submit to the output arc's inscription (color and number). Note, in some tools this inscription can be an empty function if condition is not satisfied; i.e. no tokens is produced $\{\phi\}$.

(iv) The new marking is defined as: m_k = m_j - E(A(p, t)) + (E(A(t, p)).

From the previous definitions one can see the modeling power of such formalism. The idea of defining tokens as color sets or structures means that the token is now identified since it contains data allowing differentiating it from the other tokens and it is just as any other one.

Fig 2.8 shows a simple CPN [8]. All the places are of the same color type *INT*. Places **R** and **S1** each contain one token of the same type. The transition T1 is fireable after 50 units of time and has no guard function, while T2 is immediate and has a guard function: token **m** coming from place **S2** must be grater than 10. Since **S2** contains no tokens, a token with the value of 5 is put in place O1 after 50 units of time.

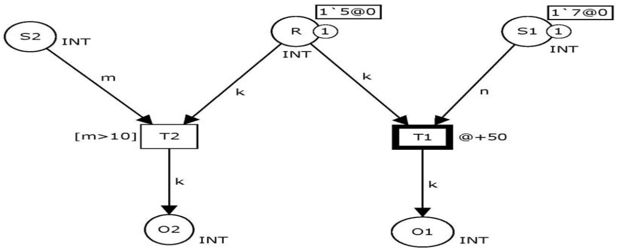


Fig. 2.8 A CPN example

Many works [9] [10] [11] were done on this formalism. However some of existing tools fall in simulation phase when the time is a main constraint. Returning to Fig. 2.8, if a token is put in place **S2** with a value greater than 10 and before the firing of T1 (during the 50 units of time), normally as the TPN definition, the transition must be fired since it is immediate. However, this is not always correct with such tool. During the simulation, both transitions can be fired which is not conform to the TPN definition.

2.6 Modeling with Object-Oriented Petri Nets

Not far from the colored Petri nets, the object-oriented Petri nets [12] [13] *OOPN* can be considered as a special kind of high level Petri nets which allow the representation and manipulation of objects. In OOPN, tokens are considered as tuples of instances of object classes which are defined as lists of attributes. It can represent all parts of complex systems, increasing the flexibility of the model. It is a collection of elements comprising constants, variables, net elements, class elements, classes, object identifiers, and method net instance identifiers.

Based on high level object oriented programming language mainly Java or C++, OOPN takes all the meanings of object programming and the characteristics of Petri nets. From this perspective, an OOPN system is composed of mutually communicating physical objects and their interconnection relations. From mathematical point of view an OOPN is defined as: ***N=(O, W)*** where:

- **O** is a set of physical objects in the system.
- **W** is a set of message passing relations among distinct objects in the system.

A physical object can be defined as $\boldsymbol{Oi= (P_i,\ T_i,\ A_i,\ M_i,\ \Sigma_i,\ G_i,\ \Lambda_i,\ E_i)}$ where M_i the input and output relationships between transitions and places for the physical object O_i. From the above definition one can find the direct relationship between the colored Petri nets and object-oriented Petri nets. If one tries to look at the OOPN we find that nearly all the characteristics of Petri nets classes are in it:

(i) Since it is a Petri net, then it inherits the ordinary Petri nets.

(ii) The timing of a transition is as the definition of a TPN.

(iii) The use of a high-level programming language enforces it with all the mathematical function found in that language, especially when talking about stochastic and random expressions.

(iv) The structured tokens makes easier the modeling of complex systems like the discrete-event systems and communication protocols.

In section 3, we will illustrate different use examples of OOPN through protocols' modules modeled with OOPN.

3 Protocol Modeling with OOPN

802.11 [14] is a wireless MAC protocol, IEEE standard, for *Wireless Local Area Network WLAN*. It is widely used in the wireless mobile internet. In 802.11, there are two mechanisms to access the medium in a fair way. The basic mechanism is the *Distributed Coordination Function DCF* [15]. It is a random access technique based on the carrier sense multiple accesses with collision avoidance (CSMA/CA) mechanism. The second mechanism to access the medium in 802.11 is the *Point Coordination Function PCF* [16] or *Priority-based access* which is a centralized MAC protocol.

When a workstation wants to transmit over 802.11 it must first sense if the channel is idle for more than a period of time called *Distributed Inter-Frame Space DIFS*. If so, it starts a random *backoff*. During the backoff time, it continues sensing the channel. If the channel stays free during the backoff, it can send its packet. However, if the channel becomes busy, it stops decrementing the backoff, but it keeps its remaining value. Then, it repeats the first step in sensing the channel to be free for more than DIFS. The last value of the backoff is restarted and decremented. Fig. 3.1 shows the access method to the channel.

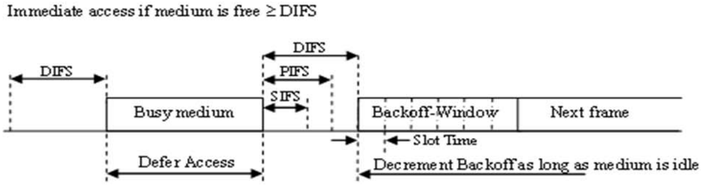

Immediate access if medium is free ≥ DIFS
DIFS
DIFS
PIFS
SIFS
Busy medium
Backoff-Window
Next frame
Slot Time
Defer Access
Decrement Backoff as long as medium is idle

Fig. 3.1 DCF access to channel

In 802.11b, a slot time equals to 20μs. *SIFS* or *Short Inter-Frame Space* equals to 10μs, and DIFS = SIFS + 2 * slot time = 10μs + 2*20μs = 50μs.

3.1 Contention Window

The value of the backoff depends on the *contention window CW* value. The workstation picks a number between zero and CW. The picked value is multiplied by the slot time to have the backoff. To decrement the backoff, the workstation continues checking the channel and each time the channel is free for a time slot, it decrements one of the picked value. However, if a collision occurs (detected by using a watchdog technique associated with the receipt of an ACK sent back by the recipient workstation) the value of CW is doubled. The minimum value of CW or CW$_{min}$ equals to 16 and the maximum value or CW$_{max}$ equals to 1024. Once a successful reception is done, the value of CW returns to CWmin.

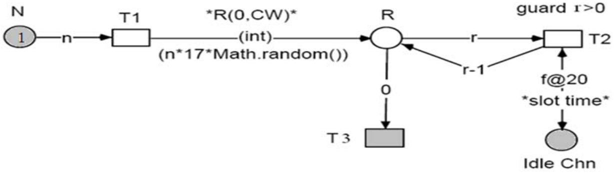


Fig. 3.2 Backoff Decrementation with OOPN

Fig. 3.2 proposes an OOPN modeling of the backoff mechanism. At the beginning the token in place N (number of transmissions) is initialized to 1. In case of collision its value is doubled. The value of N is multiplied by 16 to determine the value of CW$_{new}$. The normally distributed function *Math.random()* is used to pick a real random value between 0 and n*17 (17 is not included, and the function "int" returns an integer value between 0 and n*16). The transition T1 and T3 have no guards but T2 has a guard that must be true to be enabled which is the value of **r** must be greater than zero (the value of **r** is decremented each slot of time and the channel is always idle). Once **r** equals to zero which is a condition on the arc, T3 is enabled.

3.2 Receiving Data and Sending ACK

Once the workstation sends its packet, it waits for a time equals to SIFS and checks if it receives an acknowledgement or not. If it does not receive an acknowledgement after SIFS or 10μs, it doubles the backoff and restarts the transmission process. Fig 3.3 shows the receiving process. Since the workstation has one receive antenna, the workstation receives both ACK and data packets. It checks first if the packet belongs to it or not. The guard condition associated with transition T15 checks if the received frame is for the considered workstation. Next the guard condition of transition T10 checks if the received packet is an ACK. If it is not an ACK frame, then the T11 is fired. Hence, T10 and T11 are never in conflict and T10 is not fireable if the workstation is not the transmitter because a token must be put in place "ACK?" from the firing of T12.

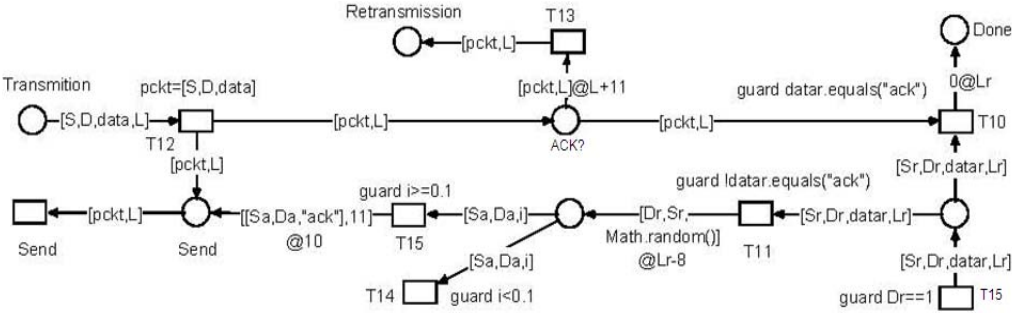


Fig. 3.3 Receiving Data

The transition T13 models a watchdog mechanism to check if the ACK is not received after a period depending on the length of the sent frame. "*L+11*" represents the time needed to transmit the data frame and a wait greater than SIFS. As in an

OOPN, the token belongs to an object class, one can define as many n-tuple based on the token attributes. As an example, the arc between place “Transmission” and transition T12 is labeled by [S, D, data, L]. This n-tuple is useful to characterize the source address of the frame (attribute S), the address of the receiver (D), the data of the frame and also the transmission time of the frame that is equivalent to its length L. From figures 3.2 and 3.3, one can see that OOPN have a modeling power comparable with TPN, SPN and CPN together.

Our approach considers two basic modules to model IEEE 802.11 network: workstation based module and medium based module. Fig.3.4 shows a detailed OOPN of a wireless workstation, modeled with “*Renew 2.1*” [17], and Fig. 3.5 for wireless medium. To design the medium module, one assumes that all workstations as potentially a bandwidth of 11 Mbps (Without considering the bandwidth attenuation which depends on the distance between two stations). The gray places and transitions in Fig. 3.5 are part of the workstations connected to the medium.

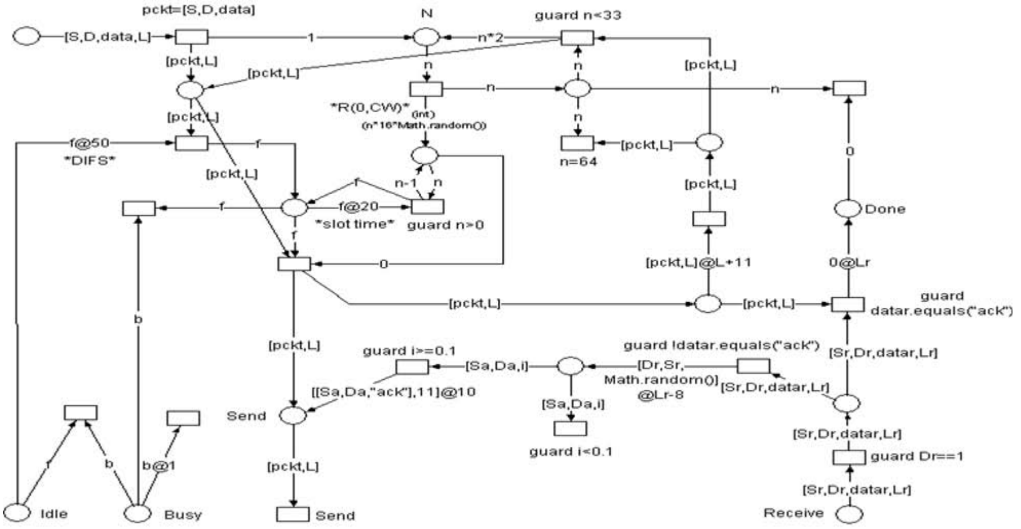


Fig. 3.4 A detailed OOPN of a Wireless Workstation

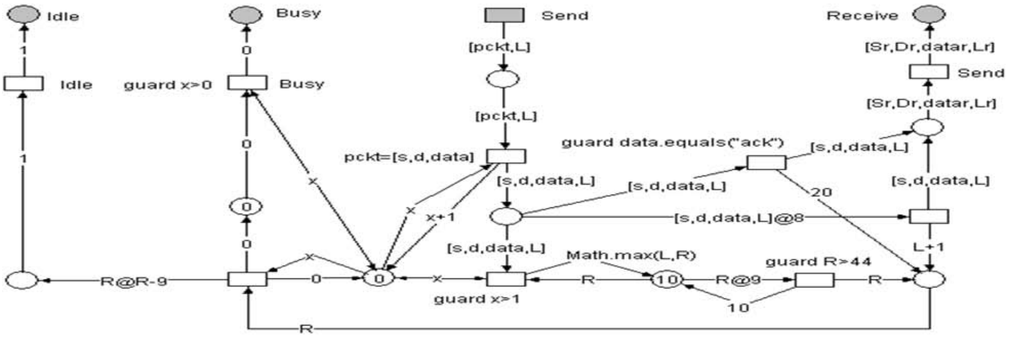


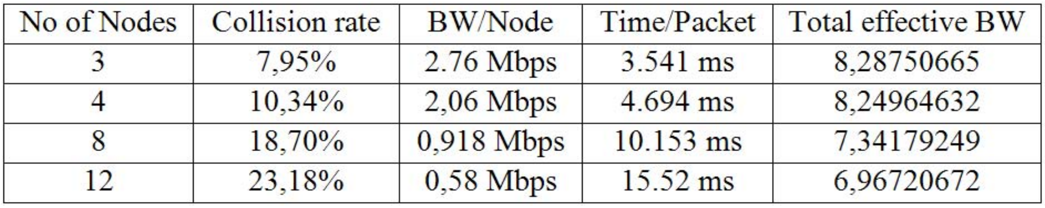
Fig. 3.5 A detailed OOPN of a Wireless Medium

3.3 Simulation and Results

Let us recall, our main objective in this study is to build protocols’ bricks to be able to evaluate DES distributed architecture. So our first goal here is to evaluate the correctness of our models of IEEE 802.11b protocols. To achieve this evaluation, we have done different simulations of adhoc architectures. The obtained results were first compared with NS2 simulations' results that we have done, Fig. 3.7. We have also compared our results with others studies' results about 802.11b adhoc architectures, see [18] and [19]. We have verified that we obtain the same results. This proves the correctness and the quality of our OOPN modeling.

Our simulations are based on dense networks with different numbers of workstations. The simulation assumes that all nodes transmit at 11Mbps and all nodes try to send data as soon as possible. Each host has 1000 packets with average length of 1150 bytes.

Table 3.1 show the simulation results:



No of Nodes	Collision rate	BW/Node	Time/Packet	Total effective BW
3	7,95%	2.76 Mbps	3.541 ms	8,28750665
4	10,34%	2,06 Mbps	4.694 ms	8,24964632
8	18,70%	0,918 Mbps	10.153 ms	7,34179249
12	23,18%	0,58 Mbps	15.52 ms	6,96720672

Table 3.1 Collision rate, Total Bandwidth and time per packet

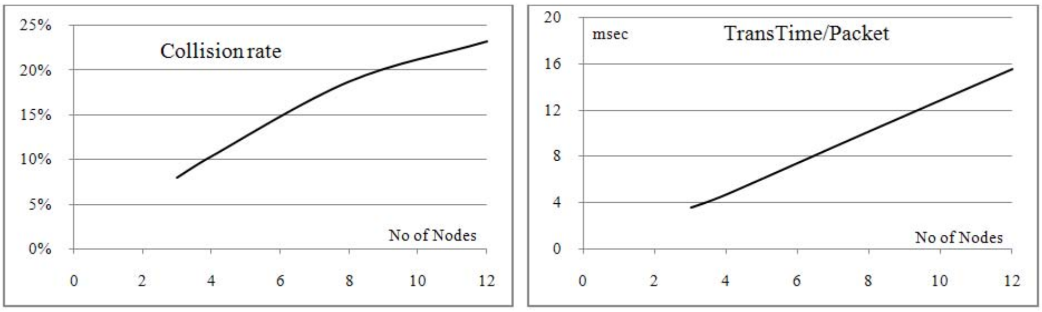


Fig 3.6 (a) collisions rate percentage

Fig 3.6 (b) time needed to transmit a packet in msec

Fig 3.6(a) shows how the collision rate increases when the number of workstations increases, while Fig. 3.6(b) shows the time needed to transmit one packet depending on the nodes on the network. Fig. 3.7 shows the throughput of 802.11b nodes sharing the 11Mbps.

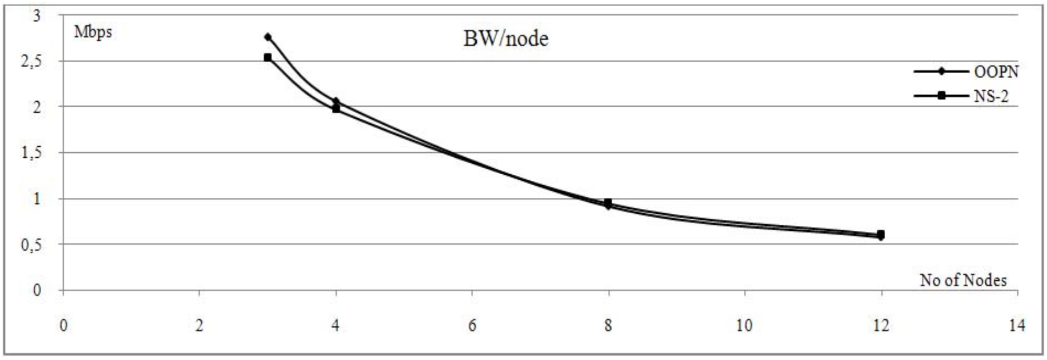


Fig. 3.7 Bit rate variation with number of nodes

4 Conclusion

In this paper we have proposed a modular OOPN approach that allows modeling in the same formalism a network protocol and the services of a DES distributed application in the future work. Let us recall here that our final goal is to be able to analyze the impact of network performances on a distributed application.

In this paper we have proved that Object-Oriented Petri Nets are well adapted to deal with all the constraints that must verify the model particularly with the possibilities to model stochastic or temporal behaviors and also to identify specific traffic. In this study, we have illustrated the capability of our approach by the simulations of IEEE 802.11b protocol and the comparisons of our results that are very closed to the values given by other studies. The modular feature of our approach allows proposing different models of same part of a system depending on the user requirement. As an example, we have shown that the medium model given here can be refined to consider the relative position of the different communicating stations. In the future, we want to propose a complete modeling framework that will allow a designer to build a model depending on the user specifications, and just by selecting the most appropriate basic models in given libraries.

References

[1] T. Murata. "*Petri nets: Properties, Analysis and Applications.*" Proc. of the IEEE, VOL 77(4), 1989.

[2] C. Ramchandani. "*Analysis of Asynchronous Concurrent Systems by Timed Petri Nets.*" Project MAC, TR120, M.I.T., 1974.

[3] P. Merlin and D. Farber. "*Recoverability of communication protocols: Implications of a theoretical study.*" IEEE Tr. Comm., VOL 24(9), 1976.

[4] B. Berthomieu, F. Peres, F. Vernadat. "*Model-checking Bounded Prioritized Time Petri Nets.*" ATVA 2007. Springer Verlag, LNCS 4762, 2007.

[5] S. Natkin. "*Les Rseaux de Petri Stochastiques et Leur Application l'valuation des Systmes Informatiques.*" PhD thesis, Cnam, France, 1980.

[6] X. Wang, L. Wang. "*WLAN System Performance Evaluate Based on SPN.*" IEEE, ICCA, 2007.

[7] K. Jensen. "*Colored Petri Nets and the Invariant-Method.*" Theoretical Computer Science, Vol. 14, 1981.

[8] http://wiki.daimi.au.dk/cpntools/_home.wiki.

[9] R. Kodikara, S. Ling, A. Zaslavsky. "*Evaluating Cross-layer Context Exchange in Mobile Ad-hoc Networks with Colored Petri Nets.*" IEEE, ICPS, 2007.

[10] W. Mata, A. Gonz?lez, R. Aquino, A. Crespo, I. Ripoll, M. Capel. "*A Wireless Networked Embedded System with a New Real-Time Kernel - PaRTiKle.*" IEEE, CERMA, 2007.

[11] L. Liu, J. Billington. "*Verification of the Capability Exchange Signalling protocol.*" STTT,VOL p (3), 2007.

[12] C. Lakos. "*From Coloured Petri Nets to Object Petri Nets.*" Lecture Notes in Computer Science, VOL 935, PATPN, 1995.

[13] Z. YU, Y. CAI. "*Object-Oriented Petri nets Based Architecture Description Language for Multi-agent Systems.*" IJCSNS, VOL 6(1), 2006.

[14] IEEE Computer Society. "*Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.*" IEEE Std. 802.11-2007.

[15] G. Bianchi. "*Performance Analysis of the IEEE 802.11 Distributed Coordination Function.*" IEEE Journal on Selected Areas in Communications, VOL 18(3), 2000.

[16] T. Suzuki, S. Tasaka. "*Performance evaluation of priority-based multimedia transmission with the PCF in an IEEE 802.11 standard wireless LAN.*" IEEE, PIMRC, 2007.

[17] http://www.informatik.uni-hamburg.de/TGI/renew/.

[18] G. Anastasi, E. Borgia, M. Conti, E. Gregori. "*IEEE 802.11b Ad Hoc Networks: Performance Measurements.*" Cluster Computing VOL 8(2-3), 2005.

[19] M. Heusse, F. Rousseau, G. Berger-Sabbatel, A. Duda. "*Performance anomaly of 802.11b.*" INFOCOM, 2003.

Composition and Independence of High-Level Net Processes[1]

H. Ehrig, K. Hoffmann, K. Gabriel, J. Padberg

Institut für Softwaretechnik und Theoretische Informatik
Technische Universität Berlin
Germany

Abstract

Mobile ad-hoc networks (MANETs) are networks of mobile devices that communicate with each other via wireless links without relying on an underlying infrastructure. To model workflows in MANETs adequately a formal technique is given by algebraic higher-order nets. For this modeling technique we here present a high-level net process semantics and results concerning composition and independence. Based on the notion of processes for low-level Petri nets we analyse in this paper high-level net processes defining the non-sequential behaviour of high-level nets. In contrast to taking low-level processes of the well known flattening construction for high-level nets our concept of high-level net processes preserves the high-level structure. The main results are the composition, equivalence and independence of high-level net processes under suitable conditions. Independence means that they can be composed in any order leading to equivalent high-level net processes which especially have the same input/output behaviour. All concepts and results are explained with a running example of a mobile ad-hoc network in the area of a university campus.

Keywords: Algebraic models, algebraic high-level nets, behavioural semantics, high-level net processes, mobility, analysis of nets, composition of processes, equivalence and independence of processes.

1 Introduction

From an abstract point of view mobile ad-hoc networks (MANETs) consist of mobile nodes which communicate with each other independently from a stable infrastructure, while the topology of the network constantly changes depending on the current position of the nodes and their availability. In our research project *Formal Modeling and Analysis of Flexible Processes in Mobile Ad-hoc Networks* we develop the modeling technique of algebraic higher-order nets. This enables the modeling of flexible workflows in MANETs and supports changes of the network topology and the subsequent transformation of workflows. Algebraic higher-order (AHO) nets are Petri nets with complex tokens, especially reconfigurable place/transition (P/T) nets in [6]. AHO-nets can be considered as a special case of algebraic high-level (AHL) nets. The main topic of this paper is to present a high-level process semantics for

[1] This work has been partly funded by the research project forMA$_l$NET (see http://tfs.cs.tu-berlin.de/formalnet/) of the German Research Council.

AHL-nets in general, where the example in Section 2 is given as a MANET and is modeled by an AHO-net.

For low-level Petri nets it is well known that processes are essential to capture their non-sequential truly concurrent behaviour (see e.g. [9,14,1,7,13]). Processes for high-level nets are often defined as processes of the low-level net which is obtained from flatting the high-level net. In [2,5] we have defined high-level net processes for high-level nets based on a suitable notion of high-level occurrence nets which are defined independently of the flattening construction. The flattening of a high-level occurrence net is in general not a low-level occurrence net due to so called assignment conflicts in the high-level net. The essential idea is to generalise the concept of occurrence nets from the low-level to the high-level case. This means that the net structure of a high-level occurrence net has similar properties like a low-level occurrence net, i.e. unitarity, conflict freeness, and acyclicity. But we have to abandon the idea that an occurrence net captures essentially one concurrent computation. Instead, a high-level occurrence net and a high-level process are intended to capture a set of different concurrent computations corresponding to different input parameters of the process. In fact, high-level processes can be considered to have a set of initial markings for the input places of the corresponding occurrence net, whereas there is only one implicit initial marking of the input places for low-level occurrence nets.

In this paper we extend the notion of high-level net processes with initial markings by a set of corresponding instantiations. An instantiation is a subnet of the flattening defining one concurrent computation of the process. The advantage is that we fix for a given initial marking a complete firing sequence where each transition fires exactly once. The main ideas and results in this paper concern the composition of high-level net processes. In general the composition of high-level net processes is not a high-level net process, because the composition may contain forward and/or backward conflicts and also the partial order might be violated. Thus we state suitable conditions, so that the composition of high-level processes leads to a high-level process. We introduce the concept of equivalence of high-level net processes, where the net structures of these high-level net processes might be different, but they have especially the same input/output behaviour. Hence their concurrent computations are compared in the sense that they start and end up with the same marking, but even corresponding dependent transitions may be fired in a different order. In this context the main problem solved in this paper is to analyse the independence of high-level net processes, i.e. under which condition high-level processes can be composed in any order leading to equivalent processes.

The paper is organised as follows. In Section 2 we exemplarily explain the concepts and results of this paper using a mobile ad-hoc network in the area of a university campus. In Section 3 on the one hand we review the notions for high-level net processes and on the other hand we introduce the new notion of high-level net processes with instantiations. In Section 4 we present our main results concerning the composition, equivalence and independence of high-level net processes. Due to space limitation the definitions and theorems are given on an informal level, while the details can be found in [4]. Finally we conclude with related work and some interesting aspects of future work in Section 5.

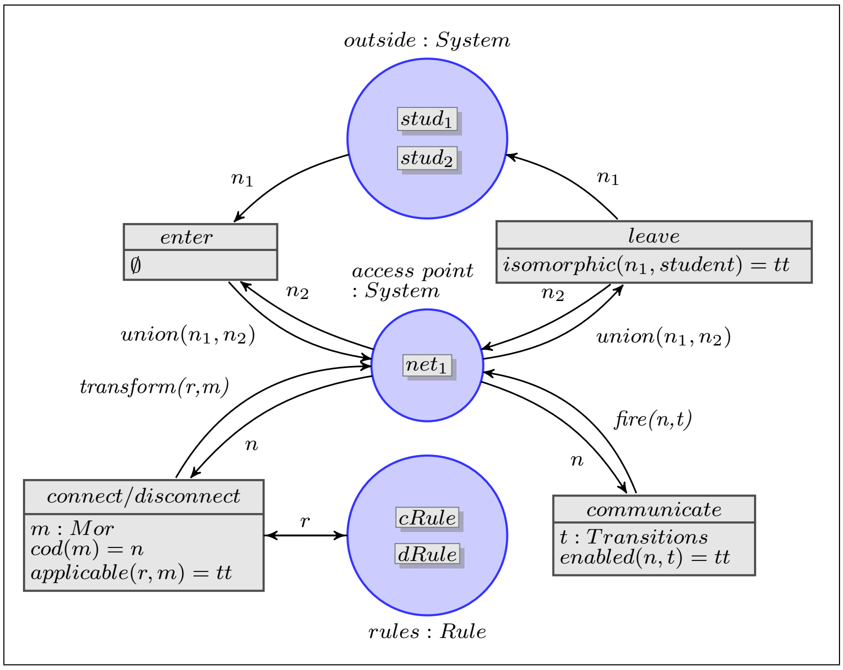


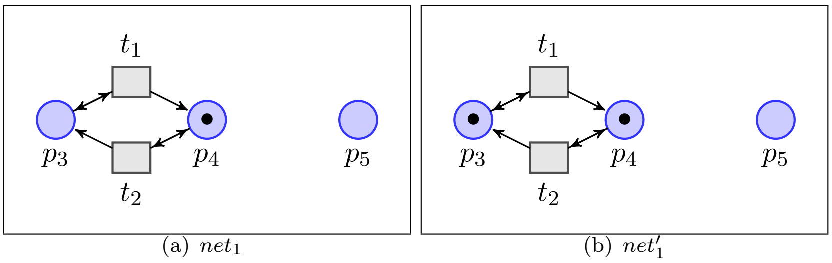
Fig. 1. AHO-net AN_{Campus}

2 Mobile Ad-Hoc Network on University Campus

In this section we introduce a simple example of a wireless network on a university campus and illustrate thereby the concepts in the following sections. As modeling technique we use algebraic higher-order (AHO) nets. AHO-nets are Petri nets with complex tokens, namely place/transition (P/T) nets and rules to support changes of the network topology. With the specific data type part in [10] they can be considered as a special case of algebraic high-level nets.

The example models a network, where students can exchange their messages. For this reason two different locations are represented by the places *outside* and *access point* in the AHO-net AN_{Campus} in Fig. 1. The marking of the AHO-net shows the distribution of the students at different places. Initially there are two students outside the campus and three additional students are on the campus represented by the tokens $stud_1$, $stud_2$ and net_1 in Fig. 1. The mobility aspect of the students is modeled by transitions termed *enter* and *leave* in Fig. 1, while the static structure of the wireless network is changed by rule-based transformations using the rules *cRule* and *dRule*. Moreover the transition *communicate* realises the well known token game.

Subsequently we concentrate on the behaviour of the transitions *communicate* and *connect/disconnect*. On the left hand side of Fig. 2 the P/T-net net_1 of the current network is depicted, where two students, represented by the places p_3 and p_4, respectively, had established a communication structure to exchange messages, while student p_5 is disconnected. The P/T-net net_1 is the token on the place *access point* in Fig. 1. To start the communication we use the transition *communicate* of the AHO-net in Fig. 1. First we give an assignment v_1 of the variables n and t in the environment of this transition and assign the network net_1 to the variable n and



(a) net_1 (b) net'_1

Fig. 2. Net tokens

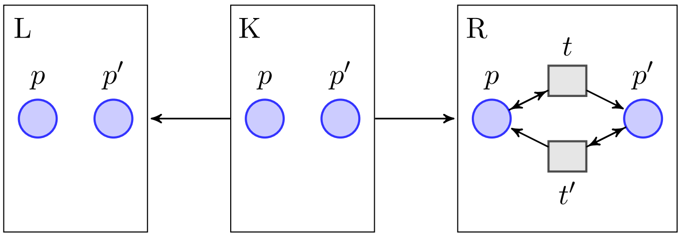
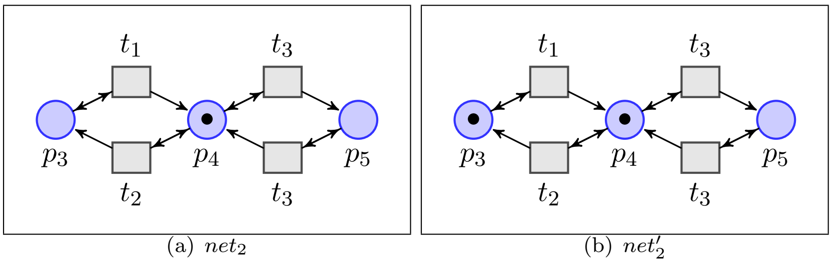


Fig. 3. Rule token *cRule*

the transition t_2 to the variable t. The firing condition checks that the student p_4 is able to send a message. This is modeled by an abstract black token on the place p_4. The evaluation of the net inscription $fire(n, t)$ realises the well-known token game by computing the follower marking of the P/T-net and so we obtain the new P/T-net net'_1 depicted on the right hand side of Fig. 2, where the student p_3 has received the message.

Assume the student p_5 wants to enter the network in order to communicate with the other students. Formally, we apply the rule *cRule* in Fig. 3 that is a token on place *rules* in Fig. 1. In general a rule $r = (L \leftarrow K \rightarrow R)$ is given by three P/T-nets called left-hand side, interface, and right-hand side respectively and the application of a rule describes the replacement of the left-hand side by the right-hand side preserving the interface. The connection between the student p_4 and p_5 is established by firing the transition *connect/disconnect* in the AHO-net in Fig. 1 using the following assignment of the variables n, r and m given in the net inscriptions of this transition: $v'_2(n) = net'_1$, $v'_2(r) = cRule$ and $v'_2(m) = g$, where g is a P/T-net morphism which identifies the left hand side of the rule *cRule* in the network $net1'$. In our case the match g maps p to p_4 and p' to p_5. The firing conditions of the transition *connect/disconnect* makes sure that on the one hand the rule is applied to the P/T-net net'_1 and on the other hand the rule is applicable with match g to this P/T-net. Finally we evaluate the term *transform(r,m)* yielding the direct transformation leading to the P/T-net net'_2 on the right hand side in Fig. 4. The effect of firing the transition *connect/disconnect* in the AHO-net in Fig. 1 with assignments of variables as discussed above is the removal of the P/T-net net'_1 from place *access point* and adding the P/T-net net'_2 to the place *access point*.

Vice versa student p_5 can enter the network net_1 by the application of the rule *cRule* to the network net_1 resulting in the network net_2 on the left hand side of Fig. 4 and afterwards students p_3 and p_4 start their communication leading to net net'_2 in Fig. 4. Formally this is achieved by firing the corresponding transitions in the



(a) net_2 (b) net'_2

Fig. 4. Net tokens after rule application

AHO-net in Fig. 1 in opposite order with suitable variable assignments v_2 and v'_1.

Summarising, we have explained two different firing sequences of the AHO-net in Fig. 1. The first one starts with the token firing of net_1 leading to the P/T-net net'_1 (see Fig. 2) before student p_5 enters the network (see right hand side of Fig. 4). The second one begins by introducing student p_5 into the network net_1 resulting in the network net_2 (see left hand side of Fig. 4) before students p_3 and p_4 exchange the message (see right hand side of Fig. 4).

Similar to processes for low-level nets we want to consider now processes for AHL-nets of which AHO-nets are a special case. These AHL-processes are based on AHL-occurrence nets. In fact the two firing sequences considered above correspond to different AHL-occurrence nets. An AHL-occurrence net is similar to a low-level occurrence net concerning unitarity, conflict freeness, and acyclicity. However, in contrast to a low-level occurrence net an AHL-occurrence net realises more than one concurrent computation depending on different initial markings and variable assignments. So we consider AHL-occurrence nets with a set of initial markings of the input places and corresponding instantiations of places and transitions by data and consistent variable assignments, respectively. For details see Section 3.

In our example we get the two AHL-occurrence nets K and K' on the left hand sides of Fig. 5 and Fig. 6 where the initial marking of the input places is given by the P/T-net net_1 and the rule *cRule*. The corresponding instantiations L_{init} and $L_{init'}$ on the right hand sides of Fig. 5 and Fig. 6 fix the two different firing sequences described above. Note that the AHL-occurrence nets K and K' have the same input and output places. But due to the firing of the transitions *communicate* and *connect/disconnect* in opposite order we use the different variable evaluations v_1 and v'_2 in L_{init} and v_2 and v'_1 in $L_{init'}$. Nevertheless the two different firing sequences end up with the same marking of the output places where the student p_5 is connected to the other students and the student p_3 received the message from student p_4 as depicted in the P/T-net net'_2 on the left hand side of Fig. 4. We show in Section 4 that there are basic AHL-occurrence nets K_1 and K_2, such that K and K' can be obtained as composition in different order of K_1 and K_2. This allows considering the corresponding processes of K and K' with instantiations as equivalent processes of the AHO-net AN_{Campus} in Fig. 1.

3 Algebraic High-Level Net Processes

In this section we review algebraic high-level nets and give a definition of high-level processes [2,5] based on high-level occurrence nets. Moreover we extend this

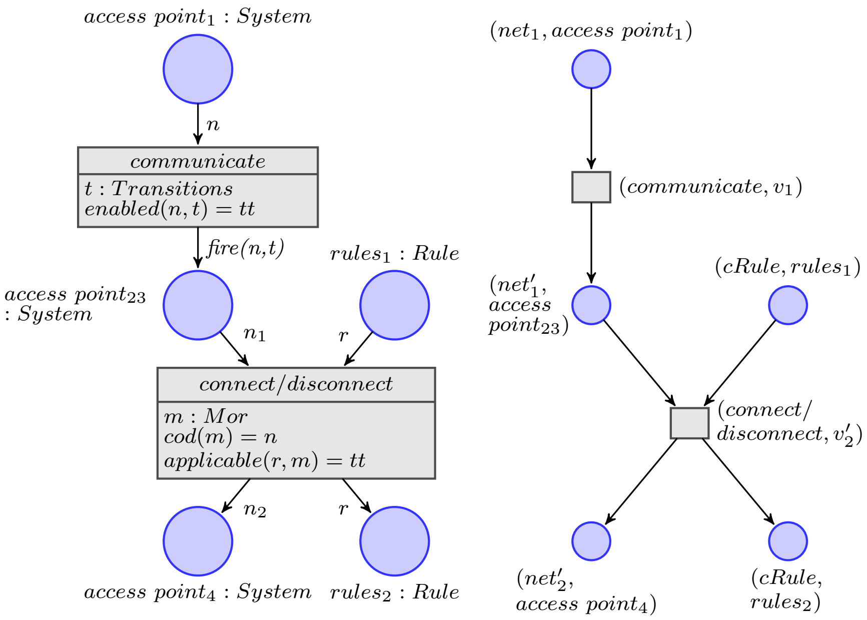


Fig. 5. AHL-occurrence net K with instantiation L_{init}

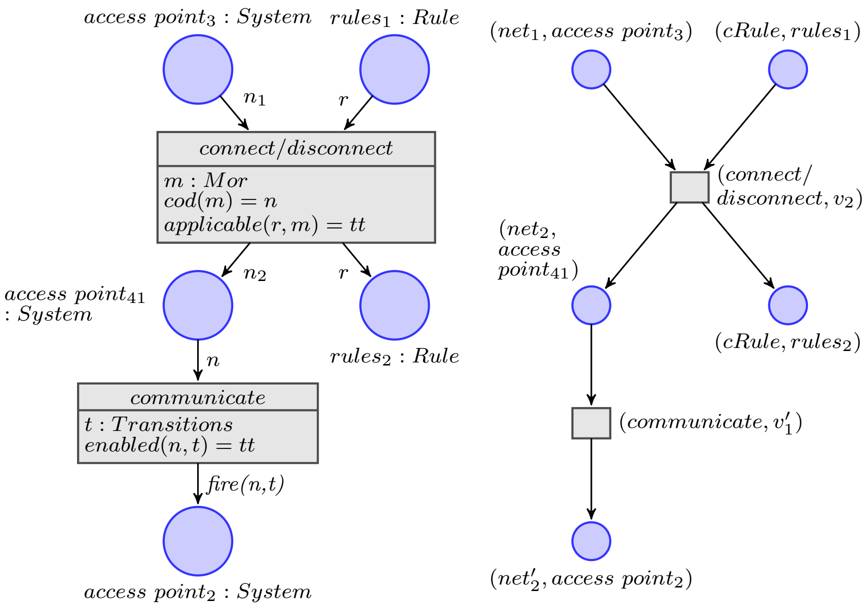
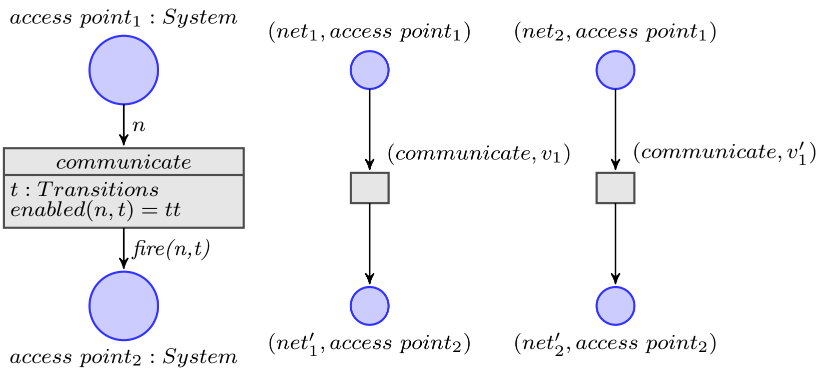


Fig. 6. AHL-occurrence net K' with instantiation $L_{init'}$

definition by a suitable notation of instantiations for each initial marking.

We use the algebraic notion of place/transition nets as in [12]. A place/transition (P/T) net $N = (P, T, pre, post)$ is given by the set of places P, the set of transitions T, and two mappings $pre, post : T \to P^{\oplus}$, the pre-domain and the post-domain, where $P^{\oplus}$ is the free commutative monoid over P that can also be considered as the set of finite multisets over P. Then we use simple homomorphisms that are generated over the set of places. These morphisms map places to places and transitions to transitions. A P/T-net morphism $f : N_1 \to N_2$ between two P/T-nets N_1 and N_2 is given by $f = (f_P, f_T)$ with functions $f_P : P_1 \to P_2$ and $f_T : T_1 \to T_2$



$access\ point_1 : System$
n
$communicate$
$t : Transitions$
$enabled(n,t) = tt$
$fire(n,t)$
$access\ point_2 : System$
$(net_1, access\ point_1)$
$(communicate, v_1)$
$(net'_1, access\ point_2)$
$(net_2, access\ point_1)$
$(communicate, v'_1)$
$(net'_2, access\ point_2)$

Fig. 7. AHL-occurrence net K_1 with instantiations L_{init_1} and $L_{init'_1}$

preserving the pre-domain as well as the post-domain of a transition. Examples of P/T nets with markings are given in Fig. 2 and Fig. 4.

An algebraic high-level (AHL) net [2,5] is essentially a P/T-net together with a suitable data type part given by an an algebraic specification and a corresponding algebra. An AHL-net morphism $f : AN_1 \to AN_2$ between two AHL-nets AN_1 and AN_2 is more or less analogously defined as a P/T-net morphism but in addition the arc inscriptions and firing conditions have to be preserved. An example of an AHL-net is given in Fig. 1. The AHO-net AN_{Campus} is a special case of an AHL-net with specific data type part defining P/T-nets and rules. For details on the signature *HLRN-System-SIG* and algebra A we refer to [10].

Now we introduce high-level occurrence nets and high-level net processes according to [2,5], called AHL-occurrence net and AHL-process respectively. The net structure of a high-level occurrence net has similar properties like a low-level occurrence net. An AHL-occurrence net K is an AHL-net such that the pre- and post domain of its transitions are sets rather than multisets and the arc-inscriptions are unary. Moreover there are no forward and backward conflicts, the partial order given by the flow relation is irreflexive and for each element in the partial order the set of its predecessors is finite.

In contrast to low-level occurrence nets a high-level occurrence net captures a set of different concurrent computations due to different initial markings. In fact, high-level occurrence nets have a set of initial markings for the input places, whereas there is only one implicit initial marking of the input places for low-level occurrence nets. The notion of high-level net processes generalises the one of low-level net processes. An AHL-process of an AHL-net AN is an AHL-net morphism $p : K \to AN$ where K is an AHL-occurrence net described above. Examples of high-level and low-level occurrence nets are given by K and K' (resp. L_{init} and $L_{init'}$) in Fig. 5 and Fig. 6.

Because in general there exist different meaningful markings of an AHL-occurrence net K, we extend this notion by a set of initial markings $INIT$ of the input places of K and a set of corresponding instantiations INS for each initial marking. An instantiation defines one concurrent execution of a marked high-level occurrence net. In more detail an instantiation is a subnet of the flattening of the AHL-occurrence net corresponding to the initial marking. The flattening $Flat(AN)$ of an AHL-net AN results in a corresponding low-level net N, where the data type part (SIG, A) and the firing behaviour of the AHL-net AN is encoded in the sets

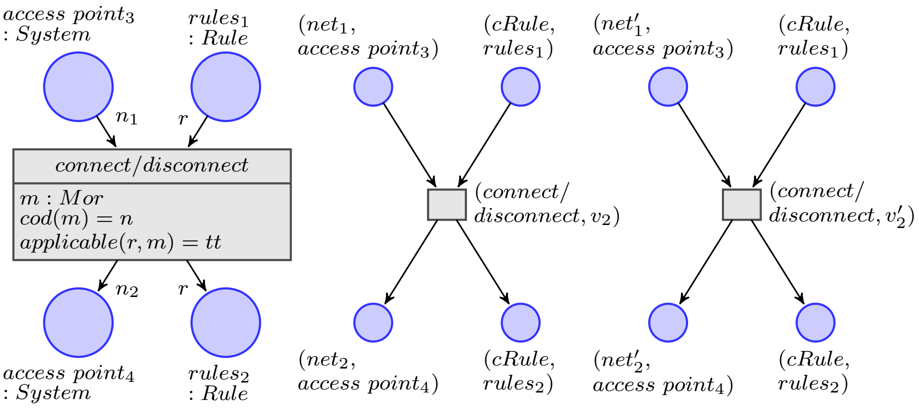

access point$_3$: System
rules$_1$: Rule
n_1
r
connect/disconnect
$m : Mor$
$cod(m) = n$
$applicable(r, m) = tt$
n_2
r
access point$_4$: System
rules$_2$: Rule
$(net_1,$ access point$_3)$
$(cRule,$ rules$_1)$
(connect/ disconnect, v_2)
$(net_2,$ access point$_4)$
$(cRule,$ rules$_2)$
$(net'_1,$ access point$_3)$
$(cRule,$ rules$_1)$
(connect/ disconnect, v'_2)
$(net'_2,$ access point$_4)$
$(cRule,$ rules$_2)$

Fig. 8. AHL-occurrence net K_2 with instantiations L_{init_2} and $L_{init'_2}$

of places and transitions of N. Thus the flattening $Flat(AN)$ leads to an infinite P/T-net N if the algebra A is infinite. In contrast the skeleton $Skel(AN)$ of an AHL-net AN is a low-level net N' preserving the net structure of the AHL-net but dropping the net inscriptions. While there is a bijective correspondence between firing sequences of the AHL-net and firing sequences of its flattening, each firing of the AHL-net implies a firing of the skeleton, but not vice versa. In [2,5] it is shown that for a marked AHL-occurrence net there exists a complete firing sequence if and only if there exists an instantiation which net structure is isomorphic to the AHL-occurrence net and has the initial marking of the AHL-occurrence net as input places.

Note that in general for a given initial marking of an AHL-occurrence net there exists more than one instantiation. Thus different firing sequences result in different markings of the output places of the AHL-occurrence net. For this reason we fix exactly one instantiation for a given initial marking, i.e. one concurrent execution of the marked AHL-occurrence net. Thus an AHL-occurrence net with instantiations $KI = (K, INIT, INS)$ is given by an AHL-occurrence net K, a set of initial markings $INIT$ and a set of corresponding instantiations INS. An instantiated AHL-process of an AHL-net AN is defined by KI together with an AHL-net morphism $mp : K \to AN$.

As an example the AHL-occurrence net with instantiations $KI_1 = (K_1, INIT_1, INS_1)$ is depicted in Fig. 7 according to the discussion in Section 2. The AHL-occurrence net K_1 is the AHL-net on the left hand side of Fig. 7. There are two different initial markings, i.e the set of initial markings is defined by $INIT_1 = \{(net_1, access\ point_1), (net_2, access\ point_1)\}$ and the set of the two instantiations on the right hand side of Fig. 7 by $INS_1 = \{L_{init_1}, L_{init'_1}\}$. The instantiated AHL-process is the AHL-occurrence net with instantiations KI_1 together with the AHL-net morphism $mp_1 : K_1 \to AN_{Campus}$. The morphism mp_1 consists of the inclusion of the transition *communicate*, while the places *access point*$_1$ and *access point*$_2$ are mapped to the place *access point* of the AHL-net AN_{Campus} in Fig. 1. Further examples are given in Fig. 5 and Fig. 6, where we have the AHL-occurrence net K with one instantiation $KI = (K, \{init\}, \{L_{init}\})$ and the AHL-occurrence net K' with instantiation KI' together with corresponding morphisms $mp : K \to AN_{Campus}$ and $mp' : K' \to AN_{Campus}$.

4 Composition, Equivalence and Independence of Algebraic High-Level Net Processes

In this section we define the composition of AHL-occurrence nets and AHL-processes with instantiations and introduce the concept of equivalence and independence of high-level net processes. The main result states that two independent high-level net processes can be composed in any order leading to equivalent high-level net processes which especially have the same input/output behaviour. For the detailed theorems and corresponding proofs we refer to [4].

The composition of two AHL-occurrence nets K_1 and K_2 is defined by merging some of the output places of K_1 with some of the input places of K_2, so that the result of the composition is an AHL-occurrence net. In general this is not necessarily true, because the result of gluing two high-level occurrence nets arbitrarily may contain forward and/or backward conflicts and may violate the partial order.

Result 1 (Composition of AHL-Occurrence Nets) *The composition of two AHL-occurrence nets K_1 and K_2 given by merging some of the output places of K_1 with some of the input places of K_2 results in an AHL-occurrence net K.*

As mentioned above instantiations define one concurrent execution of a marked AHL-occurrence net. To generalise the composition given above to the composition of instantiations we have to check that the data elements of the merged output places of K_1 and input places of K_2 are coincident in the corresponding instantiations. In this case the composition of some of the instantiations of KI_1 with some of the instantiations of KI_2 leads to suitable instantiations of the AHL-occurrence net K that is the result of the composition of the two AHL-occurrence nets K_1 and K_2.

The AHL-occurrence net with instantiations $KI_2 = (K_2, INIT_2, INS_2)$ is given in Fig. 8. The sequential composition of K_1 (see Fig. 7) and K_2 is defined by merging the output place *access point*$_2$ of K_1 and the input place *access point*$_3$ of K_2 leading to the AHL-occurrence net K (see Fig. 5). The corresponding instantiations L_{init_1} in Fig. 7 and $L_{init'_2}$ in Fig. 8 can be composed analogously to the instantiation L_{init} in Fig. 5. Note that L_{init_1} and $L_{init'_2}$ are composable, because they have the same data element net'_1 in the output and input place, respectively.

Result 2 (Composition of AHL-Occurrence Nets with Instantiations) *The composition of two AHL-occurrence nets with instantiations $KI_1 = (K_1, INIT_1, INS_1)$ and $KI_2 = (K_2, INIT_2, INS_2)$ with composable K_1, K_2 and INS_1, INS_2, respectively, is an AHL-occurrence net with instantiations $KI = (K, INIT, INS)$, where K is the composition of K_1 and K_2 and INS is the corresponding composition of INS_1 and INS_2. The set of initial markings $INIT$ is derived by the input places of the instantiations in INS.*

Given the two basic AHL-occurrence nets with instantiations KI_1 and KI_2, the composition of KI_1 and KI_2 results in the AHL-occurrence net with instantiation KI (see Fig. 5), while the opposite composition of KI_2 and KI_1 is the AHL-occurrence net with instantiation KI' (see Fig. 6).

The following result generalizes the composition to AHL-processes with instantiations where in addition the AHL-net morphisms have to be taken into account.

Result 3 (Composition of AHL-Processes with Instantiations) *Let $KI_1 = (K_1, INIT_1, INS_1)$ and $KI_2 = (K_2, INIT_2, INS_2)$ be two AHL-occurrence nets, such that $KI = (K, INIT, INS)$ is the result of their composition. Let KI_1 together with the AHL-net morphism $mp_1 : K_1 \rightarrow AN$ and KI_2 together with the AHL-net morphism $mp_2 : K_2 \rightarrow AN$ be two instantiated AHL-processes of the AHL-net AN. If the merged output places of K_1 and input places of K_2 are mapped by mp_1 and mp_2 to the same places in AN then there is one and only one AHL-net morphism $mp : K \rightarrow AN$, and KI together with the AHL-net morphism mp is an instantiated AHL-process of the AHL-net AN.*

Because for low-level occurrence nets the input/output behaviour is fixed by the net structure, two low-level occurrence nets are considered to be equivalent if they are isormorphic. For high-level occurrence nets the input/output behaviour additionally depends on the marking of their input places and on corresponding variable assignments. Hence we introduce the equivalence of two AHL-processes with instantiations, where the net structures of equivalent AHL-processes may be different, but they have the same input/output behaviour.

In more detail the AHL-occurrence nets have (up to renaming) the same sets of transitions and places and their instantiations are equivalent, i.e. there exist corresponding instantiations with the same input/output behaviour. In this case specific firing sequences of equivalent AHL-processes are comparable in the sense that they start and end up with the same data elements as marking of their input places and output places, respectively, but in general the corresponding transitions may be fired in a different order.

The AHL-processes with instantiations $KI = (K, \{init\}, \{L_{init}\})$ in Fig. 5 and $KI' = (K, \{init'\}, \{L_{init'}\})$ in Fig. 6 together with the AHL-net morphisms $mp : K \rightarrow AN_{Campus}$ and $mp' : K \rightarrow AN_{Campus}$ are equivalent. There are bijections between their transitions and places, respectively, which are not isomorphisms. The bijection of places is defined by mapping the input places of K to the input places of K' (and analogously the output places) and the place *access point*$_{23}$ of KI to the place *access point*$_{41}$ of K. Moreover the instantiations L_{init} in Fig. 5 and $L_{init'}$ in Fig. 6 are equivalent, because they have the same input and output places up to renaming.

The main result in this context are suitable conditions s.t. AHL-net processes with instantiation can be composed in any order leading to equivalent high-level net processes. Here we use especially the assumption that the instantiations are consistent, i.e. there is a close relation between their input and output places. Given the AHL-process with instantiations KI together with $mp : K \rightarrow AN$ and KI' together with $mp' : K' \rightarrow AN$ as results of the composition and opposite composition of KI_1 with $mp_1 : K_1 \rightarrow AN$ and KI_2 with $mp_2 : K_2 \rightarrow AN$. Now the question arises if KI with mp and KI' with mp' are equivalent processes.

In order to obtain equivalent processes we check that the instantiations INS_1 and INS_2 are consistent, i.e. they can be composed in any order leading to instantiations with the same input/output behaviour. Thus equivalence of KI and KI'

intuitively means that the AHL-processes KI_1 and KI_2 with consistent instantiations can be considered to be independent, because the composition in each order leads to equivalent processes.

As an example let KI_1 and KI_2 be the two instantiated AHL-processes as described above. Their sets of instantiations INS_1 and INS_2 are consistent, because the composition of the instantiations L_{init_1} (see Fig. 7) and $L_{init'_2}$ (see Fig. 8) leads to the instantiation L_{init} (see Fig. 5) and the composition of the instantiations L_{init_2} and $L_{init'_1}$ leads to the instantiation $L_{init'}$ (see Fig. 6). Thus, we state the following main result.

Main Result (Equivalence and Independence of AHL-Processes) *Given an AHL-net AN and AHL-occurrence nets $KI_1 = (K_1, INIT_1, INS_1)$ and $KI_2 = (K_2, INIT_2, INS_2)$, which are composable in both directions, with consistent instantiations and AHL-net morphisms $mp_1 : K_1 \to AN$ and $mp_2 : K_2 \to AN$. Then we have instantiated AHL-processes $KI = (K, INIT, INS)$ with $mp : K \to AN$ and $KI' = (K', INIT', INS')$ with $mp' : K' \to AN$ defined by the composition of KI_1 and KI_2 in both directions. Moreover both are equivalent processes of AN, provided that mp_1 and mp_2 are compatible with the compositions. Under these conditions KI_1 and KI_2 are called independent w.r.t. the given composition in both directions.*

Applying this main result to the AHL-net AN_{Campus} in Fig. 1 we have: The two basic instantiated processes defined by KI_1 in Fig. 7 and KI_2 in Fig. 8 are composable with consistent instantiations and the composition in both directions leads to equivalent instantiated processes defined by KI in Fig. 5 and KI' in Fig. 6. Hence the processes defined by KI_1 and KI_2 are independent.

5 Conclusion and Related Work

In this paper we have presented main results of a line of research concerning the modeling and analysis of high-level net processes. Based on the notions of high-level net processes with initial markings in [2,5] we have introduced high-level net processes with instantiations. As main results we have presented conditions for the composition and independence of high-level net processes. Under these conditions the composition of two high-level net processes leads again to a high-level net process and they can be composed in any order leading to equivalent processes. In this case the two high-level net processes are called independent.

In [8,11] the semantics of object Petri nets is defined by a suitable extension of low-level processes. Object Petri nets are high-level nets with P/T-systems as tokens. A process of an object Petri net is given by a pair of processes, a high-level net process containing low-level processes of the corresponding P/T-systems. In contrast the approach presented in this paper extends the notion of high-level net processes for algebraic high-level nets. The token structure of an algebraic high-level net is defined in its data type part that is not restricted to P/T-systems but we also use rules as tokens. Thus low-level processes of P/T-systems as tokens are not considered.

In the example of a wireless network on a university campus (see Section 2) the dynamicity of the communication structure is captured by net transformations, i.e. changes of the network topology are modeled by the application of corresponding rules. While these rules focus on modifications of the net structure, an interesting aspect of future work will be to investigate the concept of broad- and multicasting using rule-based transformations. For this reason rules to modify the marking of an AHO-net have to be introduced, so that a message can simultaneously be sent to a specific number of receivers.

Our main result of independence of high-level net processes is inspired by the results of local Church-Rosser for graph resp. net transformation [15,3], where under suitable conditions transformation steps can be performed in any order leading to the same result. In [6] we have transferred these results, so that net transformations and token firing can be executed in arbitrary order provided that certain conditions are satisfied. Further ongoing work concerns the correspondence between these different concepts of independence in more detail and transfer these results to high-level net processes.

References

[1] Degano, P., J. Meseguer and U. Montanari, *Axiomatizing Net Computations and Processes*, in: *Proc. on Logic in Computer Science (LICS)* (1989), pp. 175–185.

[2] Ehrig, H., *Behaviour and Instantiation of High-Level Petri Net Processes*, Fundamenta Informaticae **65** (2005), pp. 211–247.

[3] Ehrig, H., K. Ehrig, U. Prange and G. Taentzer, "Fundamentals of Algebraic Graph Transformation," EATCS Monographs in TCS, Springer, 2006.

[4] Ehrig, H., K. Gabriel, J. Padberg and K. Hoffmann, *Composition and Independence of High-Level Net Processes*, Technical report, TU Berlin, Fak. IV (2008), (see `http://tfs.cs.tu-berlin.de/formalnet/results.html`).

[5] Ehrig, H., K. Hoffmann, J. Padberg, P. Baldan and R. Heckel, *High-level net processes*, in: *Formal and Natural Computing*, LNCS **2300** (2002), pp. 191–219.

[6] Ehrig, H., K. Hoffmann, J. Padberg, U. Prange and C. Ermel, *Independence of net transformations and token firing in reconfigurable place/transition systems*, in: *Proc. Application and Theory of Petri Nets (ATPN)*, LNCS **4546** (2007), pp. 104–123.

[7] Engelfriet, J., *Branching Processes of Petri Nets*, Acta Informatica **28** (1991), pp. 575–591.

[8] Farwer, B. and M. Köhler, *Mobile Object-Net Systems and their Processes*, Fundam. Inform. **60** (2004), pp. 113–129.

[9] Goltz, U. and W. Reisig, *The Non-sequential Behavior of Petri Nets*, Information and Control **57** (1983), pp. 125–147.

[10] Hoffmann, K., T. Mossakowski and H. Ehrig, *High-Level Nets with Nets and Rules as Tokens*, in: *Proc. Application and Theory of Petri Nets (ATPN)*, LNCS **3536**, Springer, 2005 pp. 268–288.

[11] Köhler, M. and H. Rölke, *Reference and Value Semantics Are Equivalent for Ordinary Object Petri Nets*, in: *Proc. Application and Theory of Petri Nets (ATPN)*, LNCS **3536** (2005), pp. 309–328.

[12] Meseguer, J. and U. Montanari, *Petri Nets Are Monoids*, Information and Computation **88** (1990), pp. 105–155.

[13] Meseguer, J., U. Montanari and V. Sassone, *On the Semantics of Place/Transition Petri Nets*, Mathematical Structures in Computer Science **7** (1997), pp. 359–397.

[14] Rozenberg, G., *Behaviour of Elementary Net Systems*, in: *Petri Nets: Central Models and Their Properties, Advances in Petri Nets*, LNCS **254** (1987), pp. 60–94.

[15] Rozenberg, G., "Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations," World Scientific, 1997.

Error Analysis and Verification of an IEEE 802.11 OFDM Modem using Theorem Proving [1]

Abu Nasser Mohammed Abdullah[a,3], Behzad Akbarpour[b,4] and Sofiène Tahar[c,5]

[a] *Cadence Design Systems, Chelmsford, Massachusetts, USA* [2]

[b] *Computer Laboratory, University of Cambridge, England*

[c] *Electrical and Computer Engineering Department, Concordia University, Montreal, Quebec, Canada*

Abstract

IEEE 802.11 is a widely used technology which powers many of the digital wireless communication revolutions currently taking place. It uses OFDM (Orthogonal Frequency Division Multiplexing) in its physical layer which is an efficient way to deal with multipath, good for relatively slow time-varying channels, and robust against narrowband interference. In this paper, we formally specify and verify an implementation of the IEEE 802.11 standard physical layer based OFDM modem using the HOL (Higher Order Logic) theorem prover. The versatile expressive power of HOL helped us model the original design at all abstraction levels starting from a floating-point model to the fixed-point design and then synthesized and implemented in FPGA technology. We have been able to find a bug in one of the blocks of the design that is responsible for modulation which implementation diverts from the constellation provided in the IEEE standard specification. The paper also derives new expressions for the rounding error accumulated during ideal real to floating-point and fixed-point transitions at the algorithmic level and performs a formal error analysis for the OFDM modem in HOL.

Keywords: Formal Verification, Theorem Proving, Error Analysis, OFDM, Wireless Communication.

1 Introduction

IEEE 802.11 [14] refers to a family of IEEE standards about local area and metropolitan area wireless networks. The services and protocols specified in IEEE 802.11 map to the lower two layers, namely Data Link layer (DLL) and Physical layer (PHY) of the seven-layer OSI (Open Systems Interconnection) networking reference model. DLL consists of two sub-layers named Logical Link Control (LLC) and Media Access Control (MAC). The PHY of IEEE 802.11 is based on orthogonal frequency division multiplexing (OFDM) [23],

[1] A two pages abstract preliminary version of this work has been published as a "short paper" at FMCAD 2006 [A. N. M. Abdullah, B. Akbarpour, and S. Tahar: Formal Analysis and Verification of an OFDM Modem Design using HOL, in: Proceedings IEEE International Conference on Formal Methods in Computer-Aided Design, IEEE Computer Society Press, San Jose, California, USA, November 2006, pp. 189-190].

[2] The work was done when this author was with Concordia University.

[3] Email: nasser@cadence.com

[4] Email: ba265@cl.cam.ac.uk

[5] Email: tahar@ece.concordia.ca

a modulation technique that uses multiple carriers to mitigate the effects of multipath.

Usually, the analysis and functional verification of communications and other electronics designs, such as OFDM modems, are done using simulation. But, simulation is inadequate to check all possible inputs of a design even of moderate size and thus leaves the design partially verified. Formal verification is a technique which has proved itself as a complement to simulation to achieve a rigorous verification. Among established formal verification techniques theorem proving is particularly powerful for verifying complex systems at higher levels of abstraction.

In this paper, we use the general hierarchical methodology proposed by Akbarpour [2] for the formal modeling and verification of DSP (Digital Signal Processing) designs, to verify an implementation of the IEEE 802.11a physical layer OFDM modem [20] using HOL theorem prover [7]. The verification is performed at all levels of abstraction starting from real, floating-point, and fixed-point number systems down to Register Transfer Level (RTL) hardware implementation. For the purpose of verification, both the design specification and implementation are modeled in formal logic and then mathematical theorems are proved for correctness. We were able to find a bug in the modulation block where the constellation used in the implementation did not follow the IEEE standard specification. Besides, we derive new expressions for the round-off error accumulation while converting from one number domain to the other and carry out a formal error analysis of the OFDM modem in HOL.

The rest of the paper is organized as follows. Section 2 reviews some related work. Section 3 describes details of the OFDM modem implementation to be verified and the methodology used for verification. Section 4 describes the verification of RTL blocks of the OFDM system. Section 5 describes the error analysis of the OFDM modem and its formalization using HOL. The last section concludes the paper and provides hints for future work directions.

2 Related Work

There exist a couple of works related to the application of formal methods for the IEEE 802.11. The first one [18] models the two-way handshake mechanism of the IEEE 802.11 standard with a fixed network topology using probabilistic timed automata. Then from the probabilistic timed automaton model a finite-state Markov decision process is obtained which in turn is verified using PRISM [17], a probabilistic model checking tool. The second work [24], which identifies ways to increase the scope of application of probabilistic model checking to the 802.11 MAC (Media Access Control), presents a generalized probabilistic timed automata model optimized through an abstraction technique. Here also the results were verified using PRISM. In contrast to these related works, we focus on a completely different direction. While the first work performs model checking on an IEEE 802.11 network setting and concentrates on the protocol issues, it is concerned more with the upper layers of the OSI model than the physical layer; hence more related to software verification. The second work also uses model checking to verify the MAC protocol which resides just above the physical layer. In this paper, we concentrate only on the physical layer and its hardware implementation. Moreover, instead of model checking we use more powerful theorem proving techniques based on HOL. Besides, in the work we present here, we propose a formal error analysis of the physical layer implementation, which is to the best of our knowledge the first work of its kind to tackle this issue.

Arithmetic errors in the implementation of digital filters and the FFT algorithm have long been analysed using traditional mathematics and simulation. For instance, Tran-Thong and Liu [25] presented a detailed analysis of roundoff error in various versions of the FFT algorithm using fixed-point arithmetic. Jackson [15] analysed the roundoff noise for the cascade and parallel realizations of fixed-point digital filters. Liu and Kaneko [19, 16] presented a general approach to the error analysis problem of digital filters and FFT algorithm using floating-point arithmetic and calculated the error at the output due to the roundoff accumulation and input quantization. Error analysis is traditionally validated by comparing such theoretical results with experimental simulations. Akbarpour [2] developed a framework for the error analysis of DSP systems using the HOL theorem prover. He showed how the error analysis above, particularly those of Liu and Kaneko [19, 16], can be verified mechanically. He extended this analysis to cover floating-point and fixed-point digital filters and FFT algorithms. Akbarpour's analysis of DSP algorithms follows Harrison's verification [9] of the floating-point algorithm for the exponential function using the HOL Light theorem prover which is a prior example of formalized error analysis. In this paper, we intend to investigate error analysis in the same way as proposed by [2] but on a larger case study, here IEEE 802.11 OFDM modem. Our work proves that the approach in [2] is scalable. On the top of that, in contrast to [25, 16] which perform error analysis on single structures of FFT algorithm, we derive new expressions for the accumulation of roundoff error in IFFT-FFT combination as a computation model for the whole OFDM structure. In ideal case, the output signal of the modem should be equal to the input. But, we show that in the real implementation this is never the case because of the finite precision effects.

3 IEEE 802.11 OFDM Modem and Verification Methodology

A standard block diagram implementation of the IEEE 802.11 physical layer OFDM modem is shown in Figure 1. The first block is the random data generator, which is shown here merely for completion purpose. The next block is a quadrature amplitude modulation block (QAM). For our specific implementation, 64-QAM is used. The next block is a serial to parallel (S/P) block that can also be found in the receiver side of the block diagram. The next block is the IFFT block, one of the most important blocks of OFDM. The design uses a 64-point complex IFFT core from Xilinx Coregen Library [28]. The IFFT uses the same IP core as FFT block that comes in the receiver. The parallel to serial (P/S) circuitry makes the next block. The next block in the transmitter line is the guard interval insertion circuitry. In the receiver side, the first block is guard interval removal block. We skip to QAM demapper (DQAM) block since we discussed the other blocks before. From this block the data is serialized again and the output is received sequentially.

The design flow chosen for the OFDM modem implementation under study starts from the floating-point modeling. For this OFDM modem design, the environment used for floating-point modeling is the Signal Processing Worksystem (SPW) from Cadence [5]. The second step in the design flow is fixed-point modeling and simulation. The environment used for this purpose is the Hardware Design System (HDS), which is a set of libraries from SPW. Then VHDL codes are generated automatically for the whole system using HDS also. But, for some blocks like FFT/IFFT there was no HDS counterpart and those were imported from the Xilinx Coregen Library. Some of the VHDL codes were prepared man-

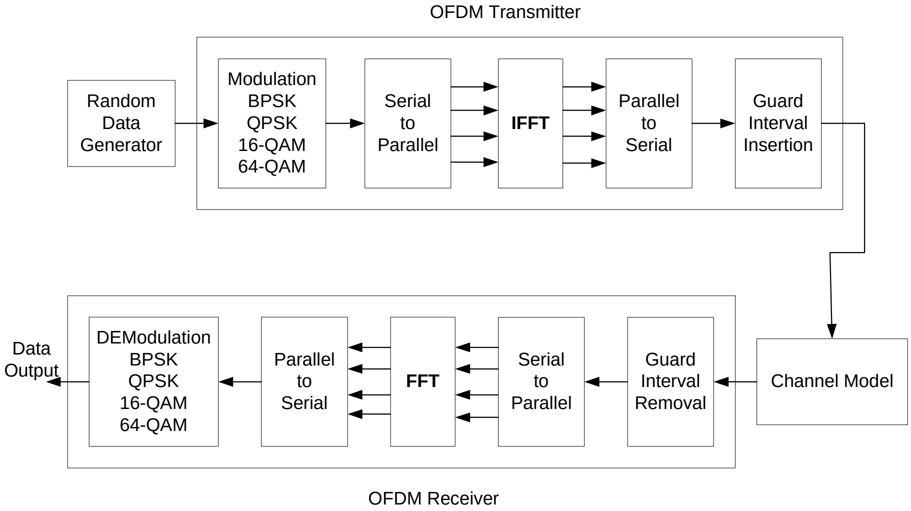


Fig. 1. OFDM Block Diagram [20]

ually [20]. After VHDL code generation, these blocks are synthesized in Synopsys Design Compiler targeting FPGA as the hardware for implementation. Finally, the synthesized circuitry is mapped into FPGA using "Place and Route" techniques and a bit file is generated.

The formal specification, verification and error analysis used in this paper is adopted

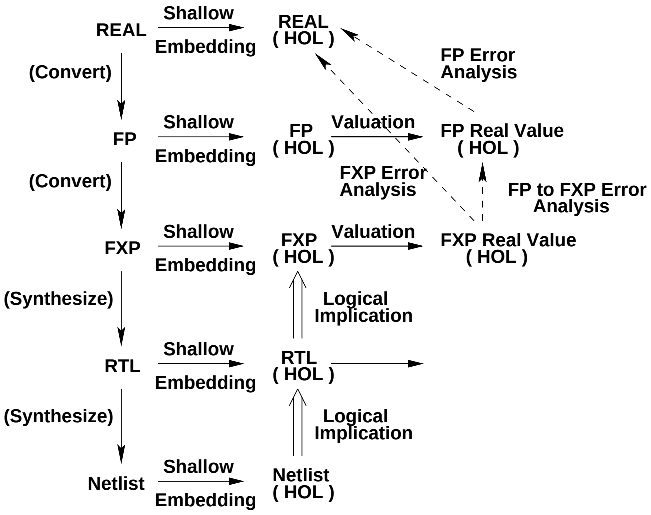


Fig. 2. DSP Specification and Verification Approach [2]

from DSP verification framework proposed by Akbarpour [2]. The commutating diagram shown in Figure 2 demonstrates the basic idea of the framework. The methodology proposes that the ideal real specification of the DSP algorithms and the corresponding floating-point (FP) and fixed-point (FXP) representations as well as the RTL (Register Transfer Level) and gate level implementations be modeled in higher order logic based on the idea of shallow embedding [4] of languages using the HOL theorem proving environment.

For the transition from real to FP and FXP levels, an error analysis is used in which the real values of the floating-point and fixed-point outputs are compared with the corresponding output of the ideal real specification. The verification of the RTL is performed using well-known classical hierarchical proof approaches in HOL. The verification can be extended, following similar manner, down to gate level netlist either in HOL or using other commercial verification tools as depicted in the figure. This analysis is not covered in this paper.

4 Formal Functional Verification

In this section we describe the verification of the RTL blocks of OFDM using HOL according to the methodology described in Section 3. The whole design is segmented into different blocks and then modeled using HOL. The resulting model is in turn set against an ideal specification and the HOL tool is used interactively to prove its correctness. In the following sections we will describe in details the verification of QAM and DQAM blocks. The details of the verification of serial to parallel (S/P) and parallel to serial (P/S) are described in Appendix A. For the blocks described below, the corresponding abstract models, HOL models and parts of the proof strategy are provided to explain the verification in its entirety. For more details please refer to [1].

4.1 Verification of QAM and DQAM Block

For the OFDM design verified, 64-QAM constellation was chosen after simulating the floating-point and fixed-point point models in Cadence SPW. The circuitry used for QAM mapping is implemented using combinational logic. It maps the input integer data into a constellation point as shown in Figure A.1. The VHDL modeling is done using a look-up table approach. The QAM block takes only 3 bits as inputs and maps to an output of 16 bits as shown in Figure A.2a. The QAM block is instantiated two times and designed to generate the real and imaginary components as two separated outputs. Each of them is formatted in 16-bit 2's complement against a 3-bit input chosen from an input of six for each block. These outputs are shown by *out_qam_r* and *out_qam_i* in Figure A.2b. The circuitry is fed by the input continuously, therefore *out_qam_r* and *out_qam_i* are generated as continuous streams. The outputs are processed in groups of 48 symbols which are stored in two separated dual port RAMs called *"Dual Port RAM image"* and *"Dual Port RAM real"*, respectively. Since, this type of RAM is generated automatically using the Xilinx Coregen Library [29] it is not discussed further. The modeling of QAM is done in HOL using different existing theories. An IF-THEN-ELSE construct is used to embed the VHDL code as shown in Table Code A.1. This model is based on the *wordTheory* [27]. The data types of VHDL can be modeled using this theory. The VHDL type *BIT* can be modeled using **T** and **F** where these represent **1** and **0**, respectively. *BIT VECTOR* can be modeled using `WORD[...]` where the dots can be replaced with any sequence of **T** or **F** separated by *";"*. As an example, bit vector "110" can be modeled as `WORD[T;T;F]`. The above model is constrained using the condition `WORDLEN input = 3` since the input is always 3 bits and thus the model does not need to be generalized for *n* bits. Here, `WORDLEN` is a function that takes any `WORD` as input and returns the length of it. We instantiated this model two times to embed the port-mapped component in HOL as *qam_mod2*.

Since the design is based on IEEE 802.11a we have used the standard [14] itself as a specification in order to verify the QAM implementation. Accordingly, for every six bits entering the *qam_mod2* block, the bits are divided into three bits each, which acts as an input to the *qam* block. Then, as described above, the *qam_mod2* block outputs two vectors containing real and imaginary parts of the modulated input. Table A.1 shows the encoding of bits for I and Q. One point can be noticed from the two tables is the similarity of bit encoding both for I and Q and this helps us to model only one specification for both. The modeling of a table in HOL is done by predicates as shown in Table Code A.2, where *I_OUT* is a triplet that accepts three bits similar to the left columns of Table A.1. For each and every argument of *I_OUT*, a unique number will be mapped as given in the tables and '$\wedge$' is used as a composition operator to construct all rows.

Having covered all the pertinent details about the implementation and a very reliable means to extract the specification, *qam_spec*, in Table Code A.3, can be written in terms of TABLES_QAM. The specification *qam_spec* is mirrored, in the same way as its implementation *qam_imp* is instantiated in *qam_mod2_imp*.

Next we will discuss the verification in details and the proof strategies adopted to bolster the correctness of the RTL implementation. The general goal is to prove that for all inputs, outputs and constraints, the QAM implementation implies the QAM specification, which can be stated in HOL as shown in Table Code A.4, where the function `TCOMP_VAL` is a simple definition based on *boolLibrary* of HOL to convert a `bool word` into its real number equivalent. We have used the existing theories of *wordTheory* and *realTheory* to build many helpful definitions and lemmas to prove the above goal and thus established the correctness of the RTL block formally. We prove the theorem and name it as *qam_imp_spec_correct*. Due to textual brevity, we do not include the whole proof procedure here line by line. Having proved this theorem it can be concluded that the QAM is formally verified. This means that the implementation conforms the specification given in the standard.

Following a similar approach we have proved the correctness of DQAM block. The details can be found in [1].

4.2 Discussion

The modeling, specification and verification done above for the OFDM RTL blocks demonstrate a way to incorporate formal methods in the verification of digital systems. We have described the implementation of the RTL blocks in HOL using formal logic. For the QAM block, it was straightforward to embed the if-then-else HDL code in HOL and the specification is obtained from the IEEE 802.11 specification. Although the demodulator block has a similar implementation and its formal description was similar to the QAM block, but finding a specification to check the design could not be done using IEEE standard since this block resides in the receiver side and the designer has the freedom to choose any way to implement it.

The main purpose for using formal verification was to find bugs in the design. We did not find any serious bug in these blocks. But, some comments are in order. Namely, for the QAM block, it is given in the standard that the input for a 64-QAM modulation must follow the constellation diagram shown in Figure A.1. The constellation gives output between -7 to 7 but the implementation used 16 bit 2's complement number to represent these numbers while 3 bits would have done the same job. If the standard is followed exactly, then this issue might have resulted in a bug in the design. But, the standard gives some flexibility to the designers in order to have more precise results from the IFFT block. As, we were aware about it at the time of verification, we constrained the implementation using the proper number of bits. The same comments are applied to the DQAM block. For the rest of the blocks we did not find any issue like this.

There are other blocks in the OFDM that we did not verify; namely, guard interval insertion and guard interval removal. The reason is that the RTL codes for those blocks were not available for the design at hand. The guard insertion block in the transmitter side has a portion of its behavioral code but the whole code mostly contains port-mapping [3] to the IP blocks. In general, the whole design contains many IP blocks and thus the verification of the design in its entirety is not practical using any theorem-proving tool like HOL. Still, this section demonstrates the scope and feasibility of formal methods in a comprehensive way in parts of the OFDM RTL blocks.

5 Formal Error Analysis

This section describes the error analysis of OFDM modem in a formal way. We first derive expressions for the accumulation of round-off error in the OFDM structure and then describe how we proved the corresponding theorems in HOL. Mainly we focus on the two computational blocks of the design—FFT and IFFT. Among all the blocks only FFT and IFFT are computational blocks doing arithmetic operation. Other blocks carry out merely mapping operations of bits from one domain to another. We take IFFT-FFT combination as the model for the error analysis of the OFDM modem. Figure 3 shows the block diagram of the IFFT-FFT combination. We first derive the equations for this system as [1].

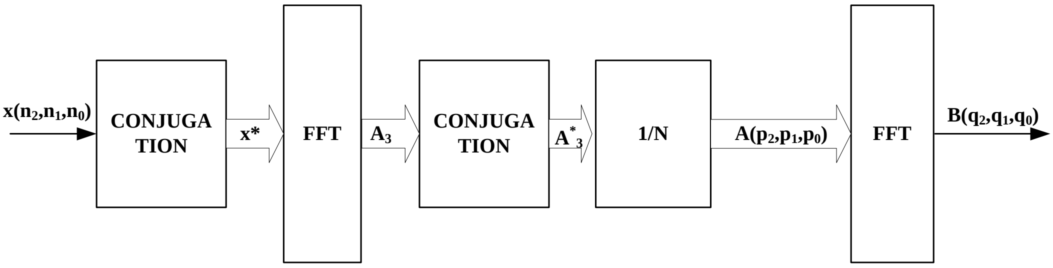


Fig. 3. Construction of IFFT-FFT

$$
\begin{aligned}
B(q_2,q_1,q_0) &= \frac{1}{64}\sum_{\mathbf{p}}\sum_{\mathbf{n}} x(n_2,n_1,n_0)(W_{64})^{(L-M)} \qquad \text{where,}\\
\sum_{\mathbf{p}} &= \sum_{p_0=0}^{3}\sum_{p_1=0}^{3}\sum_{p_2=0}^{3}\\
\sum_{\mathbf{n}} &= \sum_{n_0=0}^{3}\sum_{n_1=0}^{3}\sum_{n_2=0}^{3}\\
L &= 16q_0n_2 + (4q_1+q_0)4p_1 + (16q_2+4q_1+q_0)p_0\\
M &= 16p_0n_2 + (4p_1+p_0)4n_1 + (16p_2+4p_1+p_0)n_0
\end{aligned} \tag{1}
$$

Next we represent this mathematical model in real, floating-point and fixed-point domains. The signal $x(n)$ and twiddle factor W_{64} are complex numbers and can be written in terms of their real and imaginary components. In Equation (1) these two functions are multiplied with each other. We denote the real and imaginary parts of $x(n)$, $B(q)$, and W_{64} like C_0, D_0, C, D, U_{64}, and V_{64} and rewrite the Equation (1) as following

$$C(q_2,q_1,q_0) = \frac{1}{64}\sum_{\mathbf{p}}\sum_{\mathbf{n}} C_0(n_2,n_1,n_0)(U_{64})^{(L-M)} - D_0(n_2,n_1,n_0)(V_{64})^{(L-M)} \tag{2}$$

$$D(q_2,q_1,q_0) = \frac{1}{64}\sum_{\mathbf{p}}\sum_{\mathbf{n}} C_0(n_2,n_1,n_0)(V_{64})^{(L-M)} + D_0(n_2,n_1,n_0)(U_{64})^{(L-M)} \tag{3}$$

Mimicking the analysis of real numbers we ought to define the equations for floating-point and fixed-point number and state $fl(.)$ and $fxp(.)$ as floating-point and fixed-point, respectively. The characters prime and double primes are used to point to floating-point and fixed-point numbers and we will stick to this convention in the analysis set forth. Using these notations we denote the floating-point and fixed-point conversions of C and D as C', C'', D', D'', respectively.

In analyzing the effects of floating-point roundoff, the effect of rounding is presented multiplicatively. Let $*$ denote any of the operations $+$, $-$, $\times$, $\div$. It is known [26, 6] that if p represents the precision of the FP format, then

$$fl(x \, * \, y) = (x \, * \, y)(1 \, + \, \delta), \quad \text{where } |\delta| \, \leq \, 2^{-p}. \tag{4}$$

While the rounding error for floating-point arithmetic enters into the system multiplicatively, it is an additive component for fixed-point arithmetic. In this case, the fundamental error analysis theorem can be stated as

$$fxp(x \, * \, y) = (x \, * \, y) + \, \epsilon, \quad \text{where } |\epsilon| \, \leq \, 2^{-fracbits \; (X)} \tag{5}$$

and *fracbits* is the number of bits that are to the right of the binary point in the given FXP format X.

The real part of floating-point, C' , can be written with all the errors due to floating-point round-off as follows

$$\begin{aligned} C^{'}(q_2,q_1,q_0) \; = \; \frac{1}{64}\Bigg[\sum_{\mathbf{p}}\sum_{\mathbf{n}}\bigg(\Big(& C_0^{'}(n_2,n_1,n_0)(U_{64})^{(L-M)} \\ & (1+\delta_{1024p_2+256p_1+64p_0+16n_2+4n_1+n_0})\Big) - \\ & \Big(D_0^{'}(n_2,n_1,n_0)(V_{64})^{(L-M)} \\ & (1+\epsilon_{1024p_2+256p_1+64p_0+16n_2+4n_1+n_0})\Big)\bigg) \\ & (1+\xi_{1024p_2+256p_1+64p_0+16n_2+4n_1+n_0}) \\ & \prod_{\substack{i=1024p_2+256p_1 \\ +64p_0+16n_2 \\ +4n_1+n_0}}^{4095} (1+\lambda_i)\Bigg](1+\tau)(1+\rho) \end{aligned} \tag{6}$$

where $\boldsymbol{\delta}$ accounts for the round-off error due to multiplication of $C_0^{'}$ and $(U_{64})^{(L-M)}$ according to Equation (4). The function $\boldsymbol{\epsilon}$ represents the error due to the round-off error after the multiplication of $D_0^{'}$ and $(V_{64})^{(L-M)}$. The error due to the subtraction of $[C_0^{'}(U_{64})^{(L-M)} - D_0^{'}(V_{64})^{(L-M)}]$ is represented using $\boldsymbol{\xi}$. Based on the errors due to one single iteration, the error due to the two summations $\sum_{\mathbf{p}}\sum_{\mathbf{n}}$ (which is actually an abbreviation for six summations $\sum_{p_0=0}^{3}\sum_{p_1=0}^{3}\sum_{p_2=0}^{3}\sum_{n_0=0}^{3}\sum_{n_1=0}^{3}\sum_{n_2=0}^{3}$ can be stated as products of $\boldsymbol{\lambda}$ where the upper index is set as 4095 due to six iterations each ranging from 0 to 3 giving $4\times4\times4\times4\times4\times4-1\ = 4095$. It should have eclipsed all the rounding errors in the whole system of equation, but still the fraction $\frac{1}{64}$ incurs two round-off errors. One of them due to the division of 1 by 64, denoted as $\boldsymbol{\tau}$ and the other is for the multiplication thereafter with the rest of the equations, denoted as $\boldsymbol{\rho}$. These errors can be generalized on the same line of reasoning for the other equations.

The error related with the imaginary part $D^{'}$ of the floating-point can be written as

$$\begin{aligned} D^{'}(q_2,q_1,q_0) \ = \ \frac{1}{64}\Bigg[\sum_{\mathbf{p}}\sum_{\mathbf{n}}\Big(\Big(& C_0^{'}(n_2,n_1,n_0)(V_{64})^{(L-M)} \\ & (1+\delta^{''}_{1024p_2+256p_1+64p_0+16n_2+4n_1+n_0})\Big)- \\ & \Big(D_0^{'}(n_2,n_1,n_0)(U_{64})^{(L-M)} \\ & (1+\epsilon^{''}_{1024p_2+256p_1+64p_0+16n_2+4n_1+n_0})\Big)\Big) \\ & (1+\xi^{''}_{1024p_2+256p_1+64p_0+16n_2+4n_1+n_0}) \\ & \prod_{\substack{i=1024p_2+256p_1\\+64p_0+16n_2\\+4n_1+n_0}}^{4095}(1+\lambda_i^{''})\Bigg](1+\tau^{'})(1+\rho^{'}) \end{aligned} \tag{7}$$

where the previous function symbols used in Equation (6) are modified with double/single prime, namely $\boldsymbol{\delta}^{''}$, $\boldsymbol{\epsilon}^{''}$, $\boldsymbol{\xi}^{''}$, $\boldsymbol{\lambda}^{''}$, $\boldsymbol{\tau}^{'}$, $\boldsymbol{\rho}^{'}$; but the meaning remains the same. A point to emphasize is that all error functions are in multiplication relation with the variable and this is what makes the floating-point round-off error much complicated.

Similar formulas can be derived for the real and imaginary parts of fixed-point number, $C^{''}$ and $D^{''}$.

Adding the error parameters leaves us just one step away before we start to formalize the analysis after deriving the error that occurred in the conversion from one domain to another. We start with the real to floating-point conversion and the round-off error difference between the complex floating-point implementation and complex real implementation of IFFT-FFT denoted as $e(q_2,q_1,q_0)$. We derive the following equation that expresses the round-off error accumulated due to real to floating-point conversion,

$$e(q_2,q_1,q_0) = \frac{1}{64}\left[\sum_{\mathbf{p}}\sum_{\mathbf{n}} e_0(n_2,n_1,n_0)(W_{64})^{(L-M)} + \mathbf{f}(\mathbf{n},\mathbf{p})\right] \tag{8}$$

where we assume

$$e_0(q_2,q_1,q_0) = C_0^{'}(n_2,n_1,n_0) - C_0(n_2,n_1,n_0) + j\Big(D_0^{'}(n_2,n_1,n_0) - D_0(n_2,n_1,n_0)\Big) \tag{9}$$

and $\mathbf{f}(\mathbf{n},\mathbf{p})$ is written according to Equations (6) and (7)

$$\begin{aligned}\mathbf{f}(\mathbf{n},\mathbf{p}) = {} & C_0^{'}(n_2,n_1,n_0)(U_{64})^{(L-M)}\big[(1+\delta_{(p,n)})(1+\xi_{(p,n)})\prod_{i=(p,n)}^{4095}(1+\lambda_i)(1+\tau)-1\big] \\ & - D_0^{'}(n_2,n_1,n_0)(V_{64})^{(L-M)}\big[(1+\epsilon_{(p,n)})(1+\xi_{(p,n)})\prod_{i=(p,n)}^{4095}(1+\lambda_i)(1+\tau)-1\big] \\ & + j\Bigg[C_0^{'}(n_2,n_1,n_0)(V_{64})^{(L-M)}\big[(1+\delta^{''}_{(p,n)})(1+\xi^{''}_{(p,n)})\prod_{i=(p,n)}^{4095}(1+\lambda_i^{''})(1+\tau^{'})-1\big] \\ & - D_0^{'}(n_2,n_1,n_0)(U_{64})^{(L-M)}\big[(1+\epsilon^{''}_{(p,n)})(1+\xi^{''}_{(p,n)})\prod_{i=(p,n)}^{4095}(1+\lambda_i^{''})(1+\tau^{'})-1\big]\Bigg]\end{aligned} \tag{10}$$

The two variables n and p are used for the function as a short-hand for $n = n_2, n_1, n_0$ and $p = p_2, p_1, p_0$.

The above analysis can be adopted similarly to come at the following error function, $e^{'}(q_2,q_1,q_0)$, for the round-off error due to conversion from real to fixed-point domain

$$e^{'}(q_2,q_1,q_0) \;=\; C^{''}(q_2,q_1,q_0) - C(q_2,q_1,q_0) + j\big[D^{''}(q_2,q_1,q_0) - D(q_2,q_1,q_0)\big] \tag{11}$$

Denoting the error as $\mathbf{f}'(\mathbf{n},\mathbf{p})$, the final error can be written as

$$e^{'}(q_2,q_1,q_0) = \frac{1}{64}\left[\sum_{\mathbf{p}}\sum_{\mathbf{n}} e_0(n_2,n_1,n_0)(W_{64})^{(L-M)} + \mathbf{f}'(\mathbf{n},\mathbf{p})\right] \tag{12}$$

where $\mathbf{f}'(\mathbf{n},\mathbf{p})$ is constructed as follows

$$\begin{aligned}\mathbf{f}'(\mathbf{n},\mathbf{p}) = {} & \delta^{'}_{(p,n)} + \epsilon^{'}_{(p,n)} + \xi^{'}_{(p,n)} + \sum_{i=(p,n)}^{4095}\lambda_i^{'} + \tau^{'} \\ & + j\left[\delta^{'''}_{(p,n)} + \epsilon^{'''}_{(p,n)} + \xi^{'''}_{(p,n)} + \sum_{i=(p,n)}^{4095}\lambda_i^{'''} + \tau^{'''}\right]\end{aligned} \tag{13}$$

Equation 13 is much simplified than its real to floating-point counterpart since this error is additive but not multiplicative. To derive the errors due to floating-point to fixed-point conversion, we do not resort to derive those mammoth equations as above, rather we use the previous derivations. If the two error results derived previously are subtracted then the result gives the error we are looking for. Denoting this error as $e^{''}(q_2,q_1,q_0)$, it can be written as

$$e^{''}(q_2,q_1,q_0) = e^{'}(q_2,q_1,q_0) - e(q_2,q_1,q_0) \tag{14}$$

Figure B.1 summarizes all the error analysis discussed so far in a flow-graph format. They refer to the errors incurred in the real parts of the floating-point and fixed-point model.

5.1 Formal Error Analysis in HOL

For implementing the above error analysis in HOL, we first construct complex numbers on reals similar to [10]. We define in HOL a new type for complex numbers, to be in bijection with $\mathbb{R} \times \mathbb{R}$. The bijections are written in HOL as $complex : \mathbb{R}^2 \rightarrow \mathbb{C}$ and $coords : \mathbb{C} \rightarrow \mathbb{R}^2$. We use convenient abbreviations for the real (*Re*) and imaginary (*Im*) parts of a complex number, and also define arithmetic operations such as addition, subtraction, and multiplication on complex numbers. We overload the usual symbols $(+, -, \times)$ for $\mathbb{C}$ and $\mathbb{R}$. Similarly, we construct complex numbers on floating- and fixed-point numbers. Then we define the principal N-roots on unity ($e^{-j2\pi/N} = cos\ (2\pi n/N) - j\ sin\ (2\pi n/N)$), and its powers (*OMEGA*) as a complex number using the sine and cosine functions available in the transcendental theory of the HOL reals library [8]. We specify expressions in HOL for expansion of a natural number into a binary form in normal and rearranged order. The above enables us to specify the IFFT-FFT combination algorithm in real (*REAL_IFFT_FFT*), floating- (*FLOAT_IFFT_FFT*), and fixed-point (*FXP_IFFT_FFT*) abstraction levels using recursive definitions in HOL as described in Equation (1). Then we define the real and imaginary parts of the IFFT-FFT algorithm (*IFFT_FFT_RE*, *IFFT_FFT_IM*) and powers of the principal N-roots on unity (*OMEGA_RE*,*OMEGA_IM*). Later, we prove in separate lemmas that the real and imaginary parts of the FFT algorithm in real, floating-, and fixed-point levels can be expanded as in Equations (2) and (3). Then we prove lemmas to introduce an error in each of the arithmetic steps in real and imaginary parts of the floating- and fixed-point IFFT-FFT algorithms according to the Equations (6) and (7). We prove these lemmas using the fundamental error analysis lemmas for basic arithmetic operations [2]. Then we define in HOL the error of the *p*th element of the floating- (*REAL_TO_FLOAT_IFFT_FFT_ERROR*) and fixed-point (*REAL_TO_FXP_IFFT_FFT_ERROR*) IFFT-FFT algorithms at step q, and the corresponding error in transition from floating- to fixed-point (*FLOAT_TO_FXP_IFFT_FFT_ERROR*). Thereafter, we prove lemmas to rewrite the errors as complex numbers using the real and imaginary parts. Finally, we prove the following lemmas (Table Codes B.1,B.2,B.3) to determine the accumulation of roundoff error in floating- and fixed-point IFFT-FFT combination algorithm by recursive equations and initial conditions according to the Equations (8) to (14).

5.2 Discussion

The error analysis done above covers the OFDM rounding error analysis thoroughly between different number domains. To establish the complete theory of error analysis we proved three main theorems with the help of formalized real and imaginary part of IFFT-FFT expansion and also the theorems related to the error for arithmetic operations. All definitions were derived heavily from existing theories, e.g., *realTheory*, *boolTheory*, *ieeeTheory*, *floatTheory*, *fxpTheory*, *wordTheory*, etc. There is a very strong relationship between mathematical models and their formal counterparts which might have been observed above. The definitions built on top of established theories in turn helped to build the FFT and IFFT components; which build the theory for the IFFT-FFT combinations. Then this theory is extended and the operators are overloaded for establishing the real, floating-point and fixed-point counterparts of the design using the *floatTheory* and *fxpTheory*.

Throughout the proof of the theories built-in tactics and tacticals were used. In many of these proofs case analysis and induction were used. Our main approach to prove the theorems was to have a rough paper and pencil sketch of the approach and then formalize it using the techniques available in the HOL tool. Many times it happened that it was hard to prove the theorem as a whole in one shot and then we break the goal in manageable size to prove the parts separately to combine later. To accomplish this in a different way sometimes theorems are assumed in the proof to concentrate in the core goal and later the assumed theorem is proved. Thus we prove the theorems till the final error analysis between floating-point to fixed-point. Through the course of the modeling and proof, many lemmas are developed, some are trivial but essential and some are crucial to move to the next step in establishing a theorem. But, it is important to mention that the current theorems can be proved in a better way which is realized gradually as we moved to much complicated proofs and so the latter proofs are better and concise than the previous ones.

6 Conclusion

This paper is mainly concerned to demonstrate the use of formal verification techniques, here theorem proving, to verify an implementation of an OFDM modem based on the IEEE 802.11a physical layer standard for wireless communication. The OFDM design is fairly complex and some important design blocks were chosen for verification purposes. We formally modeled and verified the RTL blocks against the corresponding specifications in the standard. The end result showed the flawless functionality of the original implementation after abstracting the required functionality from the original design.

We also analyzed the errors in the OFDM system occurring at the time of converting from one number domain to the other, for all three domains—ideal real, floating-point, and fixed-point numbers. We used the IFFT-FFT combination as a model for the error analysis of the whole system. Then we derived new expressions for the accumulation of round-off error in the OFDM system and proved the corresponding theorems in HOL. This formalization can be considered as a large application of the formal error analysis framework described before and shows the viability of such analysis even for larger scale systems as the one analyzed.

The future work that can be carried out pertaining to this paper might elucidate new and interesting ideas and some suggestions are following:

- Verifying the RTL implementation of OFDM block using the clocking constraints.
- Development of a parameterized error analysis pattern for any FFT or IFFT design of arbitrary computing point and radix.
- Performing statistical error analysis for the OFDM modem to find average and mean square errors for IFFT-FFT combination. To perform such an analysis mechanically, we need to use a formal theory on the properties of random variables and random processes [11, 13].
- Verifying the OFDM system using a combination of HOL and another powerful computer algebra system such as Maple [21] or Mathematica [22].

References

[1] A. N. M. Abdullah. Formal Analysis and Verification of an OFDM Modem. Master's thesis, Department of ECE, Concordia University, Montreal, QC, Canada, 2006.

[2] B. Akbarpour. *Modeling and Verification of DSP Designs in HOL.* PhD thesis, Department of ECE, Concordia University, Montreal, QC, Canada, 2005.

[3] P. J. Ashenden. *Designer's Guide to VHDL.* Morgan Kaufmann, 2001.

[4] R. Boulton, A. Gordon, M. Gordon, J. Harrison, J. Herbert, and J.Van-Tassel. Experience with Embedding Hardware Description Languages in HOL. In *Theorem Provers in Circuit Design*, pages 129–156, North-Holland, 1992.

[5] Cadence Design Systems Inc. *Signal Processing Worksystems (SPW) User's Guide*, July 1999.

[6] G. Forsythe and C. B. Moler. *Computer Solution of Linear Algebraic Systems.* Prentice-Hall, 1967.

[7] M. J. C. Gordon and T. F. Melham. *Introduction to HOL: A Theorem Proving Environment for Higher-Order Logic.* Cambridge University Press, 1993.

[8] J. Harrison. Constructing The Real Numbers in HOL. *Formal Methods in System Design*, 5(1-2):35–59, 1994.

[9] J. Harrison. Floating Point Verification in HOL Light: the Exponential Function. Technical Report 428, University of Cambridge Computer Laboratory, Cambridge, UK, 1997.

[10] J. Harrison. Complex Quantifier Elimination in HOL. In *Supplemental Proceedings of Theorem Proving in Higher Order Logics*, pages 159–174. Edinburgh, UK, September 2001.

[11] O. Hasan. *Formal Probabilistic Analysis using Theorem Proving*. PhD thesis, Department of ECE, Concordia University, Montreal, QC, Canada, 2008.

[12] HOL Sourceforge Project. *The HOL System Reference.* http://hol.sourceforge.net, September 2005.

[13] J. Hurd. *Formal Verification of Probabilistic Algorithms.* PhD thesis, University of Cambridge, Cambridge, UK, 2002.

[14] IEEE 802.11 Working Group. *IEEE Std 802.11a-1999.* The Institute of Electrical and Electronics Engineers,Inc, 1999.

[15] L. B. Jackson. Roundoff-noise analysis for fixed-point digital filters realized in cascade or parallel form. *IEEE Transactions on Audio and Electroacoustics*, AU-18:107–122, June 1970.

[16] T. Kaneko and B. Liu. Accumulation of Round-Off Error in Fast Fourier Transforms. *Journal of Association for Computing Machinery*, 17(4):637–654, Oct. 1970.

[17] M. Kwiatkowska, G. Norman, and D. Parker. PRISM: Probabilistic Symbolic Model Checker. In *Computer Performance Evaluation, Modelling Techniques and Tools*, volume 2324 of *Lecture Notes in Computer Science*, pages 200–204. Springer-Verlag, 2002.

[18] M. Kwiatkowska, G. Norman, and J. Sproston. Probabilistic Model Checking of the

IEEE802.11 Wireless Local Area Network Protocol. In *Process Algebra and Probabilistic Methods, Performance Modeling and Verification*, volume 2399 of *Lecture Notes in Computer Science*, pages 169–187. Springer-Verlag, 2002.

[19] B. Liu and T. Kaneko. Error analysis of digital filters realized with floating-point arithmetic. *Proceedings of the IEEE*, 57:1735–1747, October 1969.

[20] F. Manavi. Implementation of OFDM Modem for the Physical Layer of IEEE802.11a Standard Based on XILINX VIRTEX-II FPGA. Master's thesis, Dept. of ECE, Concordia University, Montreal, QC, Canada, 2004.

[21] Maplesoft. Waterloo Maple Inc. http://www.wolfram.com/products/mathematica/index.html, 2006.

[22] Mathematica. Wolfram Research Inc. http://www.maplesoft.com/products/maple/index.aspx, 2006.

[23] R. V. Nee and R. Prasad. *OFDM for Wireless Multimedia Communications*. Artech House Publishers, 2000.

[24] A. Roy and K. Gopinath. Improved Probabilistic Models for 802.11 Protocol Verification. In *Computer Aided Verification*, LNCS 3576, pages 239–252. Springer-Verlag, 2005.

[25] T. Thong and B. Liu. Fixed-point fast fourier transform error analysis. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP 24(6):563–573, December 1976.

[26] J. H. Wilkinson. *Rounding Errors in Algebraic Processes*. Prentice-Hall, 1963.

[27] W. Wong. Modeling Bit Vectors in HOL: The Word Library. In *Higher Order Logic and Its Applications*, volume 780 of *Lecture Notes in Computer Science*, pages 371–384. Springer-Verlag, 1994.

[28] Xilinx Inc. High-performance 64-point complex fft/ifft v2.0, product specification. http://www.xilinx.com/ipcenter, 2000.

[29] Xilinx Inc. Xilinx Coregen Library. http://www.xilinx.com/ipcenter/coregen, 2005.

A Details of the Verification of RTL Blocks

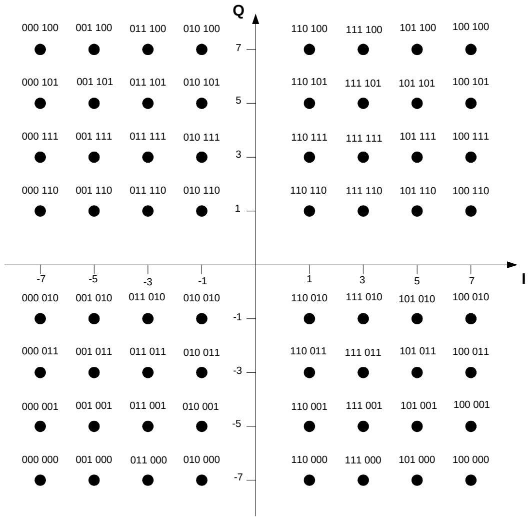
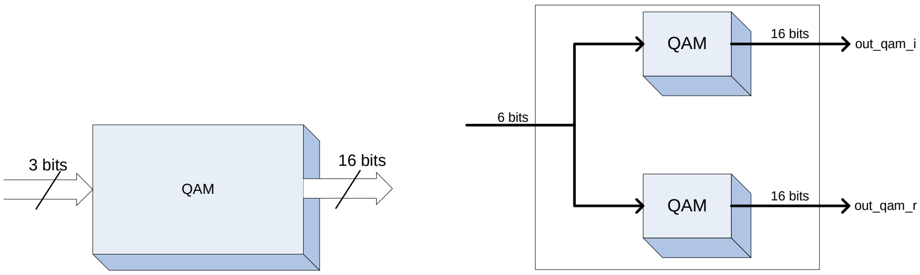


Fig. A.1. 64-QAM Constellation Bit Encoding



(a) QAM Block

(b) Instantiation of QAM Block

Fig. A.2. QAM Block and its Instantiation

Code A.1

```
⊢def  ∀input qam_out.
qam_imp (input qam_out) =
 (WORDLEN input = 3) ∧
 (if input = WORD [ F; F; F ] then
   qam_out = WORD [ T; F; F; T; F; F; F; F; F; F; F; F; F; F; F; F ]
  else
 (if input = WORD [ F; F; T ] then
   qam_out = WORD [ T; F; T; T; F; F; F; F; F; F; F; F; F; F; F; F ]
  else
 (if input = WORD [ F; T; F ] then
   qam_out = WORD [ T; T; T; T; F; F; F; F; F; F; F; F; F; F; F; F ]
  else
 (if input = WORD [ F; T; T ] then
   qam_out = WORD [ T; T; F; T; F; F; F; F; F; F; F; F; F; F; F; F ]
  else
 (if input = WORD [ T; F; F ] then
   qam_out = WORD [ F; T; T; T; F; F; F; F; F; F; F; F; F; F; F; F]
  else
 (if input = WORD [ T; F; T ] then
   qam_out = WORD [ F; T; F; T; F; F; F; F; F; F; F; F; F; F; F; F ]
  else
 (if input = WORD [ T; T; F ] then
   qam_out = WORD [ F; F; F; T; F; F; F; F; F; F; F; F; F; F; F; F ]
  else
   qam_out = WORD [ F; F; T; T; F; F; F; F; F; F; F; F; F; F; F; F ])))))))
```

Input bits (b_0,b_1,b_2)	$I - out$
000	-7
001	-5
011	-3
010	-1
110	1
111	3
101	5
100	7

Input bits (b_3,b_4,b_5)	$Q - out$
000	-7
001	-5
011	-3
010	-1
110	1
111	3
101	5
100	7

Table A.1
$64 - QAM$ Encoding Table [14]

Code A.2

```
val TABLES_QAM =
  ⊢def  ∀ I_OUT.
        TABLES_QAM (I_OUT) =
        (I_OUT (F,F,F) = ¬7) ∧ (I_OUT (T,F,F) = ¬5) ∧
        (I_OUT (T,T,F) = ¬3) ∧ (I_OUT (F,T,F) = ¬1) ∧
        (I_OUT (F,T,T) = 1) ∧ (I_OUT (T,T,T) = 3) ∧
        (I_OUT (T,F,T) = 5) ∧ (I_OUT (F,F,T) = 7)
```

Code A.3

```
 ⊢def  ∀ b0 b1 b2 I_OUT.
        qam_spec (b0 b1 b2 I_OUT) =
        ∃OUT. TABLES_QAM OUT ∧ (I_OUT b0 b1 b2 = OUT (b0,b1,b2))
```

Code A.4

```
∀ input out_qam_r out_qam_i.
   qam_mod2_imp (input out_qam_r out_qam_i) ⟹
   qam_mod2_spec input (0 b1 b2. TCOMP_VAL (WSEG 4 12 out_qam_r))
     (λ b0 b1 b2. TCOMP_VAL (WSEG 4 12 out_qam_i))
```

A.1 Verification of the S/P and P/S Blocks

A.1.1 Serial to Parallel Basics

In this section we will verify the serial to parallel block, later written as S/P, which is an indispensable part of the whole OFDM system. Most of the basics related to S/P are similar to those of the Parallel to Serial block, to be discussed later, and thus will cover almost all the important aspects of both blocks in this section. The concept of serial to parallel conversion is trivial. A long stream of data is divided into several equal or approximately equal length of chunks which can all be operated upon at the same time. From the mathematical point of view, it is the manipulation of a vector into several columns of a matrix. However, S/P conversion is very important in OFDM. The length of the blocks produced in S/P determine the number of spectral coefficients to be used by the IFFT, which is essential in choosing how many frequencies are to be used. Usually, the block length is a power of 2, which makes the IFFT and FFT algorithms most computationally efficient. Moreover, in OFDM, the data is divided among a large number of closely spaced carriers. Since the entire bandwidth is filled from a single source of data, it is necessary to transmit in a parallel way so that only a small amount of the data is carried on each carrier, and by this lowering of the bitrate per carrier, the influence of intersymbol interference is significantly reduced.

A.1.2 S/P Circuitry

The S/P circuitry is very simple to implement. It has its presence both in the transmitter and receiver of the system. In the transmitter side, it is placed between *QAM* and *IFFT* block, and in the receiver side between *Guard Removal* and *FFT* block. The design at hand has the same functionality of of *"Bits to fixp"* block of SPW [5] in fixed-point model. It consists of a shift register and a latch, which are both clocked with the same rate as the input data. Six bits from input stream are serially shifted into a register. Then they are latched for six clock cycles. There are two control signals *enable* and *clear* to synchronize the whole process.

A.1.3 S/P Modeling in HOL

Modeling of the S/P block in HOL is done in a different way than what we have seen in Section 4.1. The modeling is not exactly one to one mapping because a VHDL *PROCESS* is involved here. In fact, a *PROCESS* never terminates itself, and it can only be controlled using *WAIT* statements and sensitivity lists. After executing the last statement, a *PROCESS* will be suspended only to be resumed later on an event in the sensitivity list. This last behavior poses a difficulty in modeling it in HOL due to non-termination problem. Higher order logic is a logic of total function and it does not allow the definition of any partial function. But, there are exceptions which motivates us to define our specification for S/P in a simpler way without resorting to complex definition. For example, the following is a total and non-recursive function that uses the expressive power of HOL [12]:

```
λ x. if (? n. P (FUNPOW g n x)) then
                   FUNPOW g (@n. P (FUNPOW g n x) ∧
                        !m. m < n ==>  P (FUNPOW g m x)) x
     else ARB
```

The function `FUNPOW` is a tail recursive function from the theory *arithmeticTheory* to define function iteration. The above function does a case analysis on the iterations of function g. The finite ones return the first value at which P holds and the infinite ones are mapped to a constant named ARB that holds all the arbitrary values. ARB is a way to convert partial-functions into total functions in HOL. But, using ARB will only complicate our model without any added benefit. A VHDL *PROCESS* is more than a simple loop and we have no cases to deal with infinity rather we only have finite sets of statements to be dealt with infinitely. This discussion is to justify why we did not use certain features of HOL to model our system which seems apparently helpful in doing so. The other aspect of the model is that three signals *clk, enable,* and *clear* are not used since we are verifying this module independently of other blocks, and there are no pipelining issues involved here. Having said that we introduce the implementation of S/P in HOL as follows

```
⊢_def  ∀ cnt out_parallel input.
         Serial_Parallel_IMP (cnt out_parallel input) =
         ∃ shift_reg.
           (WORDLEN out_parallel = 6) ∧
           (shift_reg input = SHRN_bit cnt input out_parallel)
```

Apparently a simplification of the corresponding VHDL code but a little analysis will support its correct functionality. From the code, the variable cnt is a natural number whose type is defined as num; out_parallel is a bool word and input is of bool type. The implementation takes three arguments where cnt is defined to keep track of the time or bit index which is a model of the signal count. The second variable has the same name of its VHDL counterpart and so is the last one - input. A function shift_reg is defined as shift_reg:bool→bool word to mimic the VHDL signal of the same name. Variable out_parallel is constrained to six using WORDLEN function as before because the design specifies so. Since the system will receive only one input at a time and then latches all till it fills the whole shift register, so we write another definition in HOL to manipulate every new bit entering the system and filling the empty places with zeros

```
⊢_def  ∀ N M w.
         SHRN_bit (N M w) =
         WCAT (WORD (REPLICATE (WORDLEN w - (N + 1)) F),WORD [ M ])
```

This definition uses WCAT which concatenates two lists is defined in *word_baseTheory* [12] as

```
⊢_def  ∀ l1 l2. WCAT (WORD l1,WORD l2) = WORD (l1 ++ l2)
```

The symbol '++' is an infix operator that appends two lists in the above definition. The recursive definition of REPLICATE is in the theory *rich_listTheory* which replicates any variable repeatedly as specified. It is defined as

```
⊢_def  (∀ x. REPLICATE 0 x = []) ∧
         ∀ n x . REPLICATE (SUC n) x = x::REPLICATE n x
```

Here the REPLICATE function fills the rest of the places of the shift register with 'F' depending on the current value passed to it by the function and then adds the input to it. In this way at the end of the iteration the whole register will be populated with serial data and will be ready to be latched out.

Having completed the modeling of implementation we describe the specification of the block so that we can explain the verification in the next section. We state the specification of the block as

```
⊢def  ∀ t out input.
        Serial_Parallel_SPEC (t out input) = (BIT t out = input)
```

It simply puts the relation between the input and output of the block it terms of bit position. At every time `t`, we have one input entering the block which goes in the bit position related to the current index of `t` of the output. A more general approach would be to use the modulo arithmetic to model the specification, but it is not required here due to the proof strategy we followed in Section A.1.4.

A.1.4 S/P Verification

Unlike the verification strategy of QAM explained in Section 4.1, we adopt a case analysis approach to prove the goal. We can define the goal as following:

```
∀ out input t.
          (0 ≤ t ∧ t ≤ 5 ) ⟹
               Serial_Parallel_IMP (t out input) ⟹
                    Serial_Parallel_SPEC (t out input)
```

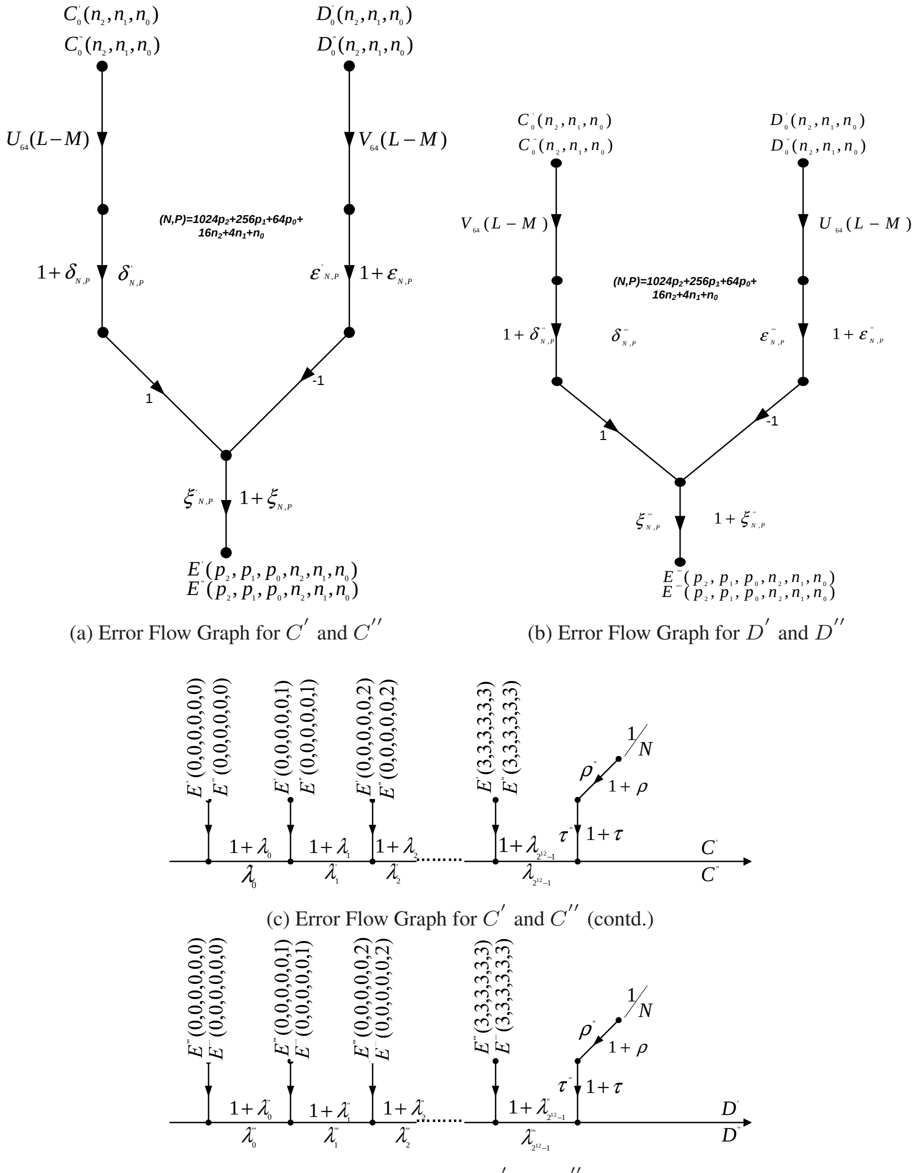
It has a very generic pattern like any other goal except the constraint which bounds t as, $0 \leq t \leq 5$. Bounding t helps to get over with the problem of looping which we stated earlier in Section A.1.3. We flatten one whole iteration which is enough to demonstrate the functional correctness of the given block. That is why we bound the variable only to check the cases starting from $t = 0$ to $t = 5$. Once we finish with case analysis we prove following trivial lemma

```
∀ t.
          (0 ≤ t ∧ t ≤ 5) ⟹
          (t = 0) ∨ (t = 1) ∨ (t = 2) ∨
          (t = 3) ∨ (t = 4) ∨ (t = 5)
```

which simply states that when t is bound between 0 and 5, then the only values for which the correctness theorem needs to hold are $t = 0, 1, 2, 3, 4, 5$. We proved the goal and thus verified the functionality of the said RTL block.

Following a similar approach, we have proved the correctness of the P/S block. The details can be found in [1].

B Details of the OFDM Error Analysis



(a) Error Flow Graph for C' and C''

(b) Error Flow Graph for D' and D''

(c) Error Flow Graph for C' and C'' (contd.)

(d) Error Flow Graph for D' and D'' (contd.)

Fig. B.1. Error Flow Graphs

Code B.1

```
∀ x q0 q1 q2. ∃ f. (IFFT_FFT_REAL_TO_FP_ERROR x q0 q1 q2 =
complex_64 * complex_sum (0,4) (λp0. complex_sum (0,4) (λp1.
complex_sum (0,4) (λp2. complex_sum (0,4) (λn0. complex_sum (0,4)
(λn1. complex_sum (0,4) (λn2.
   ERROR_0 x n2 n1 n0 * OMEGA n0 n1 n2 p0 p1 p2 q0 q1 q2 +
   f n0 n1 n2 p0 p1 p2 q0 q1 q2))))))) ∧
∃ t l d e z t′ l″ d″ e″ z″. f n0 n1 n2 p0 p1 p2 q0 q1 q2 =
complex ( Val (float_Re ((λn0 n1 n2. float_complex_round (x n0 n1
n2)) n0 n1 n2)) *
                  Val (FLOAT_OMEGA_RE n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      ((1 + d n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      (1 + z n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      mul (ER_K n0 n1 n2 p0 p1 p2,4097 − ER_K n0 n1 n2 p0 p1 p2)
                      (λi. 1 + l i) * (1 + t) − 1) −
Val (float_Im ((λn0 n1 n2. float_complex_round (x n0 n1 n2)) n0 n1 n2)) *
                  Val (FLOAT_OMEGA_IM n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      ((1 + e n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      (1 + z n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      mul (ER_K n0 n1 n2 p0 p1 p2,4097 − ER_K n0 n1 n2 p0 p1 p2)
                      (λi. 1 + l i) * (1 + t) − 1),
Val (float_Re ((λn0 n1 n2. float_complex_round (x n0 n1 n2)) n0 n1 n2)) *
                  Val (FLOAT_OMEGA_IM n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      ((1 + d″ n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      (1 + z″ n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      mul (ER_K n0 n1 n2 p0 p1 p2,4097 − ER_K n0 n1 n2 p0 p1 p2)
                      (λi. 1 + l″ i) * (1 + t′) − 1) −
Val (float_Im ((λq0 q1 q2. float_complex_round (x q0 q1 q2)) q0 q1 q2)) *
                  Val (FLOAT_OMEGA_IM n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      ((1 + e n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      (1 + z n0 n1 n2 p0 p1 p2 q0 q1 q2) *
                      mul (ER_K n0 n1 n2 p0 p1 p2,4097 − ER_K n0 n1 n2 p0 p1 p2)
                     (λi. 1 + l i) * (1 + t′) − 1))
```

Code B.2

```
∀ X M V x q0 q1 q2. ∃ f′. (IFFT_FFT_REAL_TO_FXP_ERROR X M V x q0 q1 q2 =
           complex_64 *
            complex_sum  (0,4) (λp0. complex_sum (0,4) (λp1.
            complex_sum (0,4) (λp2. complex_sum (0,4) (λn0.
            complex_sum (0,4) (λn1. complex_sum (0,4) (λn2.
              ERROR′_0 X M V x n2 n1 n0 * OMEGA n0 n1 n2 p0 p1 p2 q0 q1 q2 +
              f′ n0 n1 n2 p0 p1 p2 q0 q1 q2))))))) ∧
               ∃t′ l′ d′ e′ z′ t‴ l′ d‴ e‴ z‴.
                f′ n0 n1 n2 p0 p1 p2 q0 q1 q2 =
                   complex (d′ n0 n1 n2 p0 p1 p2 q0 q1 q2 +
                            e′ n0 n1 n2 p0 p1 p2 q0 q1 q2 +
                            z′ n0 n1 n2 p0 p1 p2 q0 q1 q2 +
                            sum (ER_K n0 n1 n2 p0 p1 p2,4096 −
                            ER_K n0 n1 n2 p0 p1 p2)(λi. l′ i) + t′,
                            d‴ n0 n1 n2 p0 p1 p2 q0 q1 q2 +
                            e‴ n0 n1 n2 p0 p1 p2 q0 q1 q2 +
                            z‴ n0 n1 n2 p0 p1 p2 q0 q1 q2 +
                            e′ n0 n1 n2 p0 p1 p2 q0 q1 q2 +
                            sum (ER_K n0 n1 n2 p0 p1 p2,4096 −
                            ER_K n0 n1 n2 p0 p1 p2) (λi. l‴ i) + t‴)
```

Code B.3

```
∀ X M V x q0 q1 q2. IFFT_FFT_FP_TO_FXP_ERROR X M V x q0 q1 q2
= right−hand side of [ REAL to FP  error theorem ] −
  right−hand side of [ REAL to FXP error theorem ]
```

The Impact of Drifting Clocks on ZigBee's Energy Efficiency

and why formal methods are pivotal to assess that

Holger Hermanns

Dependable Systems and Software
Saarland University
Saarbrücken, Germany

and

Validation de Systèmes
INRIA Rhône Alpes
Grenoble, France

Wireless embedded sensor networks are predicted to provide attractive application possibilities in industry as well as at home. IEEE 802.15.4 and ZigBee are proposed as standards for such networks with a particular focus on pairing reliability with energy efficiency, while sacrificing high data rates. IEEE 802.15.4 is configurable in many aspects, including the synchronicity of the communication, and the periodicity in which battery-powered sensors need to wake up to communicate.

In recent work, we have developed formal behavioural models of the energy implications of these options [1]. The models are modularly specified using the language MoDeST [2], which has a rigorous compositional semantics mapping on stochastic timed automata. The latter are simulated using a variant of discrete-event simulation implemented in the tool Möbius [3]. We managed to estimate energy consumptions of a number of possible communication scenarios in accordance with the standards, and derived very interesting conclusions about the energy optimal configuration of such networks. As a specific fine point, we investigated the effects of drifting clocks on the energy behavior of various application scenarios.

In this talk, I will review this joint work with Christian Groß and Reza Pulungan. I will give a detailled account of the formal modelling and analysis approach taken, and will point out why the use of formal methods appears as the only reliable as well as practical way to study such intricate questions.

References

[1] C. Groß, H. Hermanns, R. Pulungan. Does Clock Precision Influence ZigBee's Energy Consumptions?. In *OPODIS 2007*, LNCS 4878:174-188, 2007.

[2] H. C. Bohnenkamp, P. R. D'Argenio, H. Hermanns, and J.-P. Katoen. MoDeST: A compositional modeling formalism for hard and softly timed systems. *IEEE Trans.Soft.Eng.*, 32(10):812–830, 2006.

[3] D. Daly, D. D. Deavours, J. M. Doyle, P. G. Webster, and W. H. Sanders. Möbius: An extensible tool for performance and dependability modeling. In *TOOLS 2000*, LNCS 1786:332–336, 2000.

Performance Evaluation of Sensor Medium Access Control Protocol Using Coloured Petri Nets

Mohammad Abdollahi Azgomi and Ali Khalili

Department of Computer Engineering,
Iran University of Science and Technology, Tehran, Iran
azgomi@iust.ac.ir and al_khalili@comp.iust.ac.ir

Abstract. Formal modelling techniques can be used for analysis of wireless sensor networks (WSNs). Coloured Petri nets (CPNs) that is an extension of Petri nets is a powerful modelling technique. In this paper, we present a CPN model for modelling and performance evaluation of a medium access control protocol in WSNs named sensor-medium access control protocol (S-MAC). S-MAC is an energy-aware MAC protocol with nodes scheduling. The proposed model for this protocol uses the hierarchical modelling capability of CPNs. By using CPNs in this case study and the proposed method for modelling packet broadcast, we have demonstrated the possibility of modelling and evaluation of any other MAC protocol in WSNs or mobile ad-hoc networks (MANET).

1 Introduction

Wireless sensor networks (WSNs) consist of a set of small, cheap, and low-power sensor nodes that use wireless technology for communications. Comparing to other types of networks, WSNs have some notable limitations, such as processing ability, memory capacity and battery life-time. As a result of these limitations and the requirements for some new protocols, much research is engaged in this field [1].

The general approach for analysis of WSNs is to use the existing simulation tools or simulators, such as OPNET [2], NS-2 [3] and etc. For using these tools, we must consider that the result of simulating an algorithm may be different, depending on the selected tool, because of important divergence between simulators [4]. An alternative to simulation is to employ formal modelling and analysis techniques. Using these techniques, both performance evaluation and model checking can be performed. While this approach is widely used in traditional networks, advanced WSNs algorithms and protocols present a set of challenges to be formally modelled by the existing methods and tools, some of which are addressed in [5].

Coloured Petri nets (CPNs) [6], which are an extension of Petri nets, are an appropriate modelling language. CPNs have a graphical notation that is based on an underlying mathematical definition and provide several analysis methods, including simulation, state space analysis and invariant analysis. A major benefit of using CPNs is to obtain complete and unambiguous specifications of system behaviour.

As a case study, we have used CPNs and CPN Tools [14] for performance evaluation of sensor-medium access control protocol (S-MAC). The results are presented in this paper.

The remainder of the paper is organized as follows. In section 2, we report the related works. In section 3, an informal description of S-MAC protocol is provided. Section 4, describes the modelling approach used in the case study and the consequent results are given in section 5. Finally, some concluding remarks are mentioned in section 6.

2 Related Works

A lot of results are reported in literature, which have used formal techniques for modelling and analysis of traditional networks. In [7], Petri nets are used for modelling IEEE802.3 protocol in a traditional local area network (LAN) to construct a detailed model. The model is used to evaluate some performance measures. In [8], a high-level Petri net named finite population queuing system Petri nets (FPQSPN) is introduced for modelling and simulation of medium access control layer in computer networks.

Given the increasing sophistication of WSN algorithms and the difficulty of modifying an algorithm once the network is deployed, there is a clear need to use formal methods to validate system performance or functionality prior to implementing such algorithms [5]. In published works, different approaches have been used for specifying and modelling sensor networks. In these works, based on the requirements for performance evaluation or model checking purposes, a modelling technique has been selected and employed. Some new extensions of Petri nets are also proposed for these purposes, which use some extra information in places [9] or dynamic configuration capability [10] in Petri nets structure. Such extensions, attempt to extend Petri nets to model dynamic behaviour of WSNs, such as mobility of nodes, node death (as a result of battery limitation) or node failures. In [11], Petri nets are used for modelling and simulation of a routing protocol in a mobile ad-hoc network (MANET). In this work, a topology approximation mechanism is proposed to address mobility problem and performs simulation to show that this mechanism can indeed mimic the mobility of a MANET.

3 Sensor Medium Access Control Protocol

Since we will use the specification and operations of S-MAC in the next sections, we will briefly review this protocol in the following paragraphs, which is based on [13].

The open system interconnection (OSI) model defines a layered architecture for network protocols. The medium access control (MAC) layer is responsible for determining which node is allowed to access underlying layer (i.e. physical medium) at any moment. One fundamental task of a MAC protocol is to avoid simultaneous transmissions (i.e. collision) and the basic mechanism used for reducing the possibility of collision in contention based MAC is referred to as carrier sense, multiple access (CSMA). In this mechanism each node, before starting a transmission, senses the medium to find it clear and then starts its transmission.

In wireless environment, IEEE802.11 protocol [12] is a family of standards which introduces a number of MAC schemes and the physical (PHY) layer for WLAN. The primary MAC scheme of the standard is called distributed coordination function (DCF) and has two variants: basic access (BA) and request to send/clear to send (RTS/CTS). The BA scheme (that is also known as two-way handshaking scheme), has the simplicity advantage, but the possibility of (large) data packet is somewhat high and thus can cause to waste energy and degrade performance. In RTS/CTS scheme, 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 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 replies with a CTS packet. 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.

The IEEE802.11 is used in some wireless sensor networks. Considering the attributes and limitations of WSNs, we need some special protocols that designed for them. In WSNs, one of the important requirements is efficient usage of energy to prolong network lifetime and must be considered in any aspect of communication protocols, routing algorithms and query processing approach. The so-called S-MAC scheme [13] is designed to reduce energy consumption of communicating nodes. It is based on a simple observation that for most WSN applications, the sensed data streams are generated at low frequency (there is nothing to be sensed at most of the time.).

Designing S-MAC aims at reducing energy consumption from all sources of energy waste (i.e. idle listening, collision, overhearing and control overhead). The protocol is based on two distinct operational states for 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 operational 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 SYNC packets between neighbouring nodes. A SYNC packet contains the sender's chosen schedule. Each node either chooses its own schedule or follows a schedule received from one of its neighbours. As soon as a node picks a schedule, it broadcasts it so that all neighbours can update their table. Although the aim of coordinated sleeping is to synchronise neighbouring nodes on a single and shared schedule, it is possible for neighbouring nodes to have different schedules. This happens whenever a node that has announced its own schedule receives a different schedule from one of its neighbours. Anyhow, a node may receive different schedules from its neighbours and in this case, it must behave based on its schedule table for sending a packet to any node.

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. SYNC). 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. This technique with the aid of network allocation vector (NAV) variable can solve the hidden terminal problem.

4 Modelling S-MAC with CPNs

In this section, we describe our proposed coloured Petri net (CPN) model for S-MAC. The global model has been constructed taking the advantage of the hierarchical capabilities of CPNs [6]. Hierarchical CPN allows the construction of a large model as a set of smaller models connected to each other using well-defined interfaces (i.e. substitution transitions). In this way, a complex model, like the one we will present in this section, can be reduced to constructing some smaller models. The model aims to be as detailed as possible (i.e. it tries to model all operations of the MAC layer and scheduling of nodes).

Like [13], topology of the model is a two-hop network with two sources and two sinks, as shown in Fig. 1. Packets from the sources, A and B, flow through the node C and will end at sinks, D and E. Each node has a buffer containing received packets. For simplicity, we assumed that the size of the buffer is one packet. In addition, the collision may occur in network. Consider that this collision can occur only in control packets (i.e. SYNC or RTS/CTS) and data packets do not collide. We assume that the transmission media is ideal and noise-free and thus, every sent packet will receive to its destination node. It may also encounter a collision. We have also assumed that there is not any queueing and backoff delay as in [13]. We assumed that the propagation and processing delays can be ignored, too. In this case, only carrier sense delay, transmission delay and sleep delay are taken into account. Consider that transmission delay depends on packet size and is a constant value for each type of packet and thus large packets (e.g. data packets) have a larger transmission delay than small packets (e.g. control packets).

To describe the model, we will first explain the main part of the model and higher level of the model hierarchy. Then submodels will be described. For constructing and analysing the model, we have used the CPN Tools [14], which is a well-known and powerful tool for modelling and analysis of CPNs.

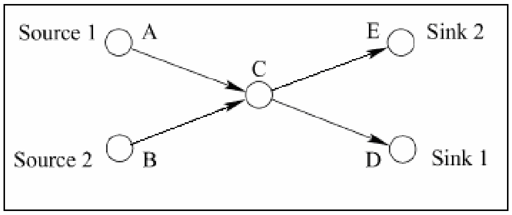


Fig. 1. Topology of the model, two-hop network with two sources and two sinks.

4.1 CPN Model for Wireless Channel

In wireless media model, if a node sends a packet, the media must broadcast it to all nodes. Each node can "hear" the packet if and only if the source of the packet is one of the neighbours. In this model, when a node transmits a packet, the packet is delivered by all nodes according to their sequence numbers. When a node gets a packet, checks the source of the packet and if it is a neighbour node, it will "hear" the packet. Anyhow, the packet is given to the channel for delivering to the next node. If the packet is delivered by all nodes, the channel drops the packet. Because there is no propagation delay, this operation must be performed without any time consumption and all nodes will deliver the packet at the same time. Here, we assume that nodes have not any movement, but considering this approach for modelling broadcast behaviour, we can have mobile nodes (e.g. in mobile wireless sensor or ad-hoc network). In mobile networks, neighbouring detection is performed dynamically based on physical location of nodes and radio transmission range.

Fig. 2 shows model of the wireless channel and network nodes as substitution transitions for a network with two nodes. The hierarchical capability of CPNs provides scalability of the model in the network.

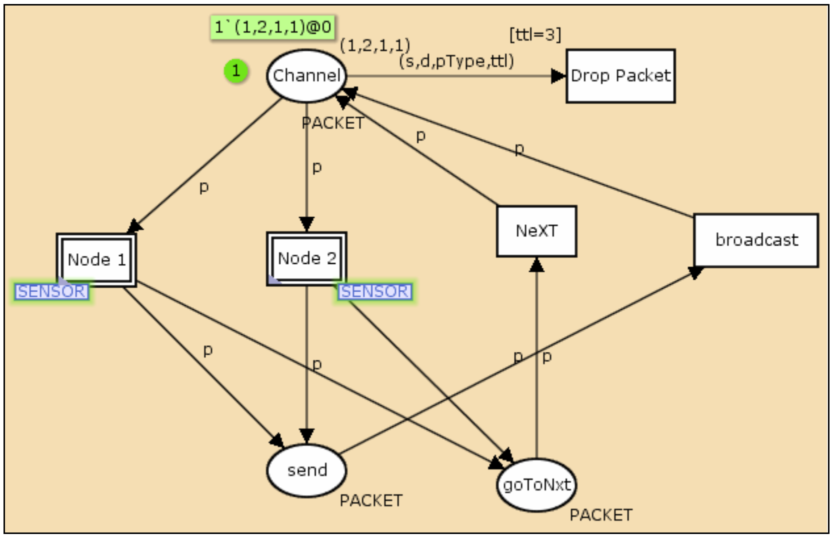


Fig. 2. Main part of wireless channel model with only two nodes (nodes are modelled as substitution transitions).

4.2 CPN Model of Nodes

In this subsection we describe CPN submodels for node scheduling, table of neighbours scheduling, sending a message and listening to channel.

Node Scheduling. In S-MAC protocol, each node has a constant sleep and wakeup period. First (and after a small random period of time), each node chooses his own scheduling and announces it to its neighbours using a SYNC message. The SYNC message shows the beginning of the wakeup time. In sleep mode, nodes turn off the radio to save energy. In wakeup period, a node for transmitting a packet must sense the channel and if the channel is free (physically and virtually), it sends RTS packet and waits for CTS confirmation. Then, it sends the data packet and waits for the acknowledgment. The basic part of the model is shown in Fig. 3. The right side of the figure illustrates neighbours schedule table and the left side, models the synchronisation procedure.

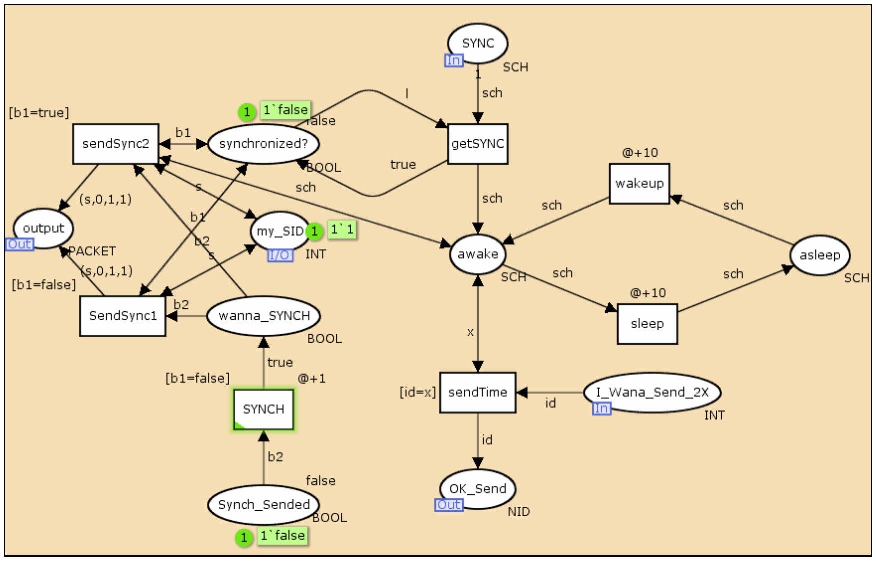


Fig. 3. Basic part of node scheduling model.

Table of Neighbours Scheduling. In each node, there exists a table that contains scheduling information of its neighbours. Model of this table is illustrated in right side of Fig. 3. When a SYNC packet is received by a node, the required information can be obtained from the packet for updating schedule table. By assuming predefined and constant values for sleep and wakeup periods, each node can update and maintain this table independently and achieve target node scheduling at any time.

Sending a Message. Transmitting a message to a neighbour node is performed based on neighbours schedule table. In this case, after determining channel state, if the channel is free, the packet will be sent based on the information retrieved from schedule table. Fig. 4 shows the basic part of message transmission model.

4.3 Hierarchical Model

The hierarchical structure of the composed CPN model is illustrated in Fig. 6. The wireless channel model is in the highest level of the model hierarchy that models the physical layer and broadcast communications. In the next level, models of network nodes exists that model basic behaviour of the nodes and in the lowest level, there are some submodels that model the nodes behaviour in detail.

In general, using hierarchical structure of CPNs facilitates modelling complicated and large systems. In addition, this capability provides scalability to easily change the topology of the network and the number of nodes.

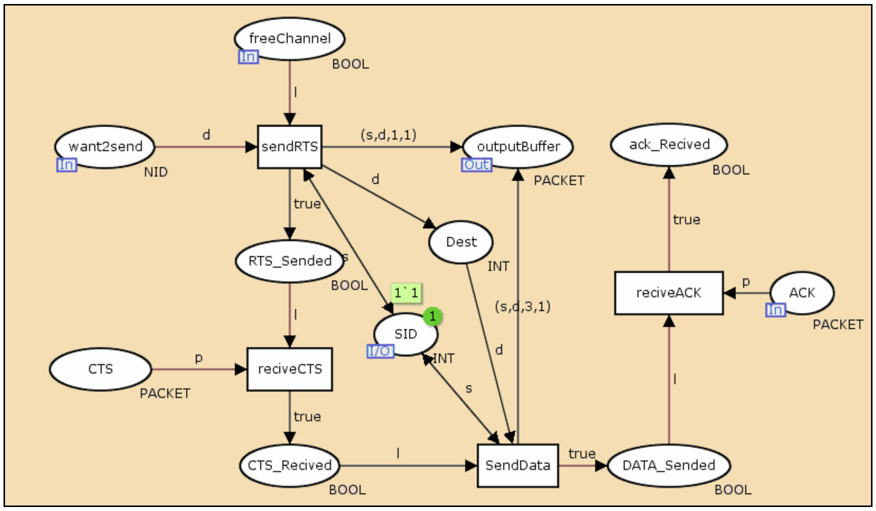


Fig. 4. The basic part of message transmission model that shows the message transmission procedure stages (channel state retrieval, sending RTS, receiving CTS and sending ACK).

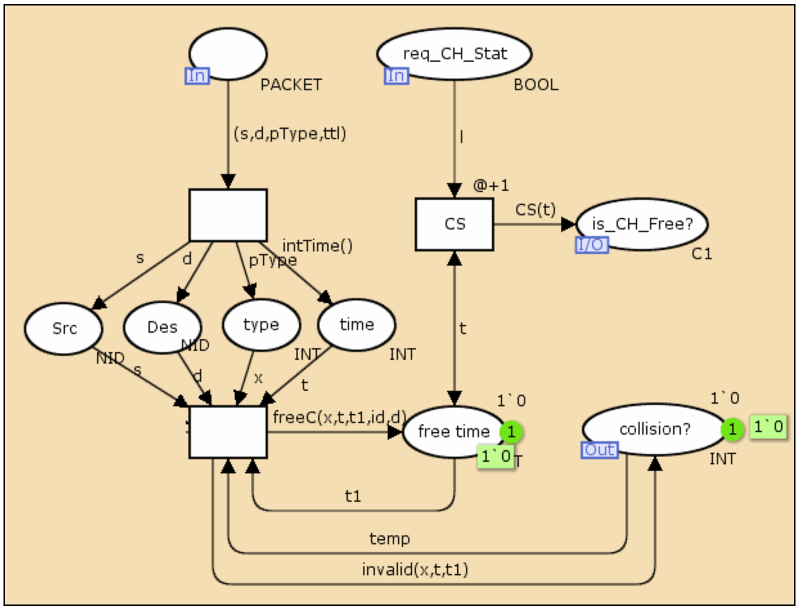


Fig. 5.The basic part of listening to the channel model.

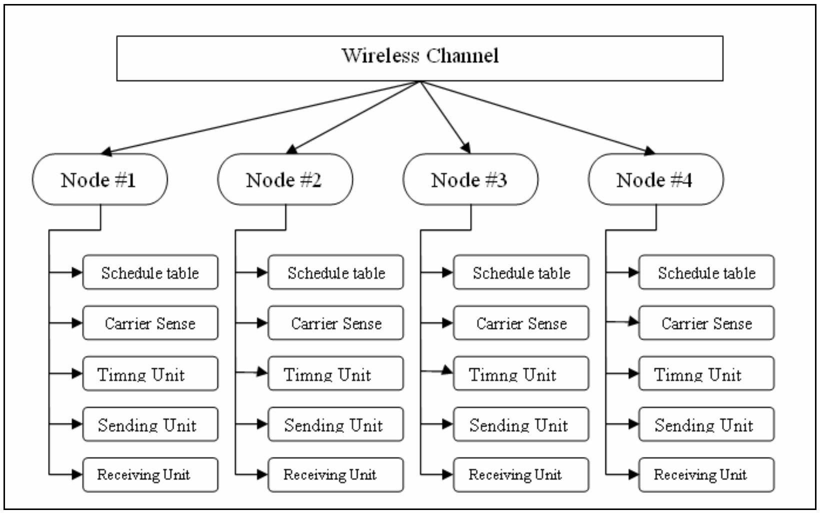


Fig. 6. Hierarchical architecture of the CPN model (for a network with four nodes).

5 Simulation Results

After constructing the model, we can extract the related performance measures using model simulation. In CPN Tools, this goal can be achieved using the monitor mechanism. A monitor is a mechanism in CPN Tools that is used to observe, inspect, control, or modify the simulation of a CPN model and the desired measures. In our study of S-MAC protocol, performance measures include *energy consumption* and *packet delivery delay*. For example, result of the model simulation for average energy consumption in source nodes for transmission of some constant data packets versus message inter-arrival time with duty cycle 20% and 50% is shown in Fig. 7. More measures may be defined and computed in a similar way.

6 Conclusions

In this paper, we presented a CPN model for S-MAC protocol. S-MAC is a medium access control protocol introduced for wireless sensor networks with the aim of reducing energy consumption. Using the hierarchical capability of CPNs, we have modelled a wireless environment (wireless media and network nodes). Then, we have evaluated some performance measures by using CPN Tools and its simulation features.

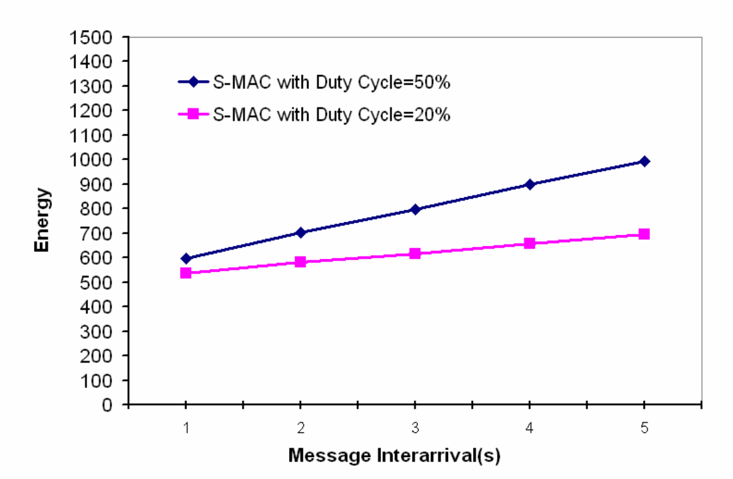


Fig. 7. Simulation results for average energy consumption in source nodes versus message inter-arrival time.

Results of this case study can help the design process of a real network. Modelling enough details in this model, shows the capabilities of CPNs for modelling and evaluation of wireless sensor and ad-hoc networks. Using hierarchical CPNs to model WSNs and its protocols have two main advantages that are flexibility and scalability of model construction.

As a future work, we intend to construct a comprehensive model for mobile sensor networks. We are attempting to add a routing protocol to this model by employing the hierarchical capability of CPNs.

References

1. Akyildiz, I.A., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A Survey of Sensor Networks, IEEE Communication Magazine, Vol. 40 (2002) 102–115.
2. OPNET Modeler, URL: http://www.opnet.com/products/modeler/home.html.
3. ns-2 network simulator, URL: http://www.isi.edu/nsnam/ns/.
4. Cavin, D., Sasson, Y., Schiper, A.: On the Accuracy of MANET Simulators, Proc. of POMC, ACM Press (2002) 38–43.
5. Ölveczky, P.C., Thorvaldsen, S.: Formal Modeling and Analysis of the OGDC Wireless Sensor Network Algorithm in Real-Time Maude, Lecture Notes in Computer Science , Vol. 448 , Springer Berlin Heidelberg (2007) 1611-3349.
6. Jensen, K.: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use, Volumes 1-3, Basic Concepts. Monographs in Theoretical Computer Science, Springer-Verlag (1997).
7. Morera, P.H., Gonzalez, T.M.P.: A CPN Model of the MAC Layer. In K.Jensen (ed.): Proceedings of the 2nd Workshop on Practical Use of Coloured Petri Nets and Design/CPN, Aarhus (1999) 153-172

8. Čapek, J.: Petri Net Simulation of Non-deterministic MAC Layers of Computer Communication Networks, Ph.D. Thesis, Czech Technical University (2003).
9. Luo, Y., Tsai, J.J.P.: A Graphical Simulation System for Modeling and Analysis of Sensor Networks, Proc. of the 7th IEEE Int'l Symp. on Multimedia (2005).
10. Graff, C.J., Giardina, C.: Dynamic Petri Nets: A New Modelling Technique for Sensor and Distributed Concurrent Systems, Proc. of the IEEE Military Communications Conf. (2005).
11. Xiong, C., Murata, T., Tsai, J.: Modeling and Simulation of Routing Protocol for Mobile Ad hoc Networks Using Coloured Petri Nets, Proc. of Workshop on Formal Methods Applied to Defense Systems (2002) 145-153.
12. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification, IEEE Std. 802.11 (1999).
13. Ye, W., Heidemann, J., Estrin, D.: An Energy-Efficient MAC Protocol for Wireless Sensor Networks, IEEE/ACM transaction on networking, Vol. 12, No. 3 (2004).
14. CPN Tools, URL: http://wiki.daimi.au.dk/cpntools/.

Validating Wireless Congestion Control and Reliability Protocols using ProB and Rodin

Jens Bendisposto, Michael Jastram, Michael Leuschel
Christian Lochert, Björn Scheuermann, Ingo Weigelt[1,2]

Computer Science Department
Heinrich Heine University
Düsseldorf, Germany

Abstract

Implicit hop-by-hop congestion control is a novel congestion control paradigm for wireless multihop networks. Implemented in the CXCC protocol, it has already proven its performance in simulations and measurements. Since CXCC makes extensive use of the properties of an inherently unreliable medium, it is, however, vitally necessary to validate the correctness of the protocol. Indeed, an early version of the CXCC protocol contained deadlocks. CXCC is complemented by an end-to-end reliability mechanism in the BarRel transport protocol. In combination, both protocols offer TCP-equivalent service in dynamic wireless multihop environments, including, e.g., route changes. BarRel relies on properties of CXCC. It therefore likewise deserves validation.
In this work we attempt to validate the CXCC and BarRel protocols using formal methods. To this end, we are developing various models in B and Event-B. We are using the ProB tool to animate and model check the formal models and the Rodin platform to formally prove correctness properties. In this paper we present first encouraging steps towards a full formal validation of the protocols.

Keywords: Protocol Validation, Wireless Networks, Congestion Control, End-to-End Reliability, CXCC, BarRel, B-Method, ProB, Rodin

1 Implicit Hop-by-Hop Congestion Control

In previous work we have introduced the Cooperative Cross-layer Congestion Control (CXCC) protocol and the Backpressure Reliability (BarRel) protocol. CXCC is a congestion control approach, and BarRel is the corresponding end-to-end reliability mechanism. Both protocols in combination provide TCP-equivalent end-to-end service. In this section, we briefly outline CXCC and BarRel. For a more detailed discussion, we refer the reader to [6, 7].

1.1 CXCC

CXCC is a cross-layer approach, encompassing essentially the MAC and transport layers. It uses implicitly obtained information for hop-by-hop backpressure conges-

[1] Email: {bendisposto,jasteam,leuschel,lochert,scheuermann}@cs.uni-duesseldorf.de

[2] The authors are grateful to the German Research Foundation (DFG) and the EU funded FP7 research project 214158 DEPLOY for partially funding this research.

tion control and single-hop reliability. The key concept is that, for each end-to-end connection, an intermediate node may only forward a packet towards the destination after its successor along the route has forwarded the previous one. This yields a backpressure mechanism which reacts very rapidly to varying medium conditions and effectively avoids excessive packet inflow into congested network areas.

The CXCC protocol realizes the implicit hop-by-hop congestion control principle by overhearing the medium after a transmission. No further packets for the same connection may be transmitted, until the packet has been forwarded further by the downstream node. The connection is "blocked" at the first node until the second node forwards the packet further on (see Fig. A.1 in appendix). In such a setting, overhearing the transmission of the downstream node can serve a dual purpose: it constitutes an implicit acknowledgment, indicating the successful reception of the previous packet by the downstream node, and at the same time is a signal to unblock the connection, allowing the next packet to be transmitted.

Link layer acknowledgments as in IEEE 802.11 are, consequently, not necessary. Just at the last hop the packet is therefore acknowledged explicitly. However, wireless communication is very error-prone. Transmissions may be lost, for instance, due to a collision. CXCC uses Request For Acknowledgment (RFA) packets in order to overcome such situations. An RFA is a small control packet, containing just enough information to identify the packet it refers to. Upon reception of an RFA, a node checks whether it already has received the respective data packet or not. It may then provide appropriate feedback, such that, if necessary, the transmission is repeated. For details see [7].

1.2 BarRel

The CXCC retransmission mechanisms allow to overcome individual transmission errors between adjacent nodes along the route. Because CXCC limits the number of packets per hop to one, there will, by construction, also be no packet loss due to buffer overflows. Hence, there will be no packet loss with CXCC as long as the route to the destination remains stable. In this situation, a source node can implicitly obtain information about successful packet deliveries to the destination node: there is at most one packet per hop in CXCC, and there is neither packet loss nor packet reodering in the network. Consequently, if the route to the destination is currently n hops long, after the $(i+n)$-th packet of a transmission has left the source node, the i-th packet must have arrived at the destination.

In BarRel, this is used in order to realize TCP-equivalent reliable end-to-end data transport without the need for a continuous stream of end-to-end acknowledgment traffic. In wireless multihop environments, such acknowledgment traffic is considered a major problem, because it is forwarded over the same wireless medium as the data traffic (in opposite direction), causing significant overhead and frequent collisions. Therefore, an ACK-free reliable transport protocol is highly desirable.

However, it may also happen that a wireless link fails permanently. Then, the above assumptions do not necessarily hold true, and packet loss *may* happen. If one node along the route is no longer reachable, the routing protocol used in the network will discover a new route. In order to guarantee end-to-end reliability also in case of link breaks and re-routing, the BarRel mechanism steps into the breach. When re-routing occurs, a BarRel source node will go back to the first packet for which it could not yet confirm successful delivery, and retransmit from there.

The delivery of the last n packets of a packet burst can not be acknowledged as described above. BarRel therefore includes further mechanisms to confirm their arrival at the destination. For this, it possible to add a number of *empty* packets at the end of the transmission. By the abovementioned implicit delivery confirmation mechanism, when the n-th such "capacity refill" (CaRe) packet leaves the source node, it is known that the last data packet must have successfully arrived. This allows for a protocol operation without any oncoming end-to-end control traffic. For details see [6].

2 Validating CXCC and BarRel

The B-method [1] is a formal methodology for the systematic development of safety-critical software systems, based on the idea of refinement. Event-B (e.g., [4]) is the successor of the B-method, which is also suitable to model reactive systems. Both are supported by industrial-strength proving tools. Event-B now spurts the Rodin platform [2] which enables integrated editing and proof. Our ProB tool [5] can be used to animate B and Event-B models, as well as validate temporal logic formulas via model checking.

In this paper, we decided to use the Event-B method to verify the correctness of BarRel and CXCC; in future work we also plan to investigate the use of the CSP process algebra (which is also supported by our tool ProB). Our goal is to validate the CXCC and BarRel protocols using animation and model checking, but also to try and develop a version of the algorithm which is correct by design. We also want to evaluate if the B Method and the tools ProB [5] and Rodin [2] are appropriate for the design of networks protocols. For the verification we developed two different sets of formal models. The first version was a set of models used for animation and model checking rather than for proof. The goal was to get familiar with the problems that arise from the domain. Also we wanted to gain confidence in the correctness of the protocols. The second model is used to obtain correctness by design, by refining a very abstract model towards the CXCC/BarRel protocols.

2.1 First version: models for animation

We specified CXCC and BarRel in two different models. The CXCC model contains a cycle free sequence of nodes, the route, for an end-to-end connection. The nodes are equipped with two buffers, one for incoming packets and one to buffer packets sent. The model can put packets into on end of the route and remove them from the other end. During transmission we can lose packets and acknowledgements and it is also possible that the implicit acknowledgment gets lost. The model does not yet cope with multiple packets on multiple routes. We also developed a graphical representation that could be used to demonstrate the protocol to domain experts who are not familiar with the mathematical notation. The absences of deadlocks was validated using ProB, as were several LTL formulas specifying correct delivery under certain fairness assumptions.

2.2 Second version: models for proof

A flow in a CXCC enabled network can be abstractly seen as a queue. The sender adds data packets to it and the receiver removes them. Another constraint resulting from CXCC and BarRel is that we can divide this queue into different contiguous sections. Each section contains only data-packets, CaRe-packets or duplicates of received data-packets. Furthermore CaRe-packets can only form the last section of the queue and duplicates precede data.

The first model m0 defines a generic Queue, mainly modeled after [3]. Methods to enqueue, dequeue and delete the whole Queue have been implemented. This queue will then subsequently be refined until the detailed behaviour of CXCC and BarRel is obtained.

m0 is then refined to map the queue elements to links along the route holding data packets. Accordingly the maximum size of the Queue is limited to the routelength. The events enqueue and dequeue now represent sending and receiving data by the source and receiver. A routebreak refines the delete event to additionally reset the counter used by the sender. As another step towards realistic network conditions the routelength is now nondeterministically changed by this event.

The next refinement step is to introduce CaRe packets. At this refinement step the queue section for CaRe packets is simply represented by a number holding the number of packets in it. Accordingly two new events sendCaRe and receiveCaRe are modeled that simply incremend and decrement respectively this value.

The routebreak event in m1 removes all elements in the queue and resets the senders counter to retransmit exactly the lost packets. In m3, we remove `routelength` many packets. This reflects, that the sender must assume the worst case. As a result, the receiver now has to cope with duplicate packets arriving.

An animation of the last model with ProB can be seen in Fig. B.1 in the appendix for referees. As far as proving is concerned, m0 and m3 still contain one unproven proof obligation (POs) each (out of 57 and 7 resp.). Models m1 and m2 are fully proven, where 88 of 94 and 69 of 78 POs were automatically proven.

References

[1] Abrial, J.-R., "The B-Book," Cambridge University Press, 1996.

[2] Abrial, J.-R., M. Butler and S. Hallerstede, *An open extensible tool environment for Event-B.*, in: *ICFEM06*, LNCS 4260 (2006), pp. 588–605.

[3] Abrial, J.-R. and D. Cansell, *Formal construction of a non-blocking concurrent queue algorithm (a case study in atomicity)*, Journal of Universal Computer Science **11** (2005), pp. 744–770.

[4] Abrial, J.-R., D. Cansell and D. Méry, *Refinement and reachability in Event$_b$*, in: H. Treharne, S. King, M. C. Henson and S. A. Schneider, editors, *ZB*, Lecture Notes in Computer Science **3455** (2005), pp. 222–241.

[5] Leuschel, M. and M. Butler, *ProB: A model checker for B*, in: K. Araki, S. Gnesi and D. Mandrioli, editors, *FME 2003: Formal Methods*, LNCS 2805 (2003), pp. 855–874.

[6] Scheuermann, B., "Reading Between the Packets – Implicit Feedback in Wireless Multihop Networks," Ph.D. thesis, Heinrich Heine University, Düsseldorf, Germany (2007).

[7] Scheuermann, B., C. Lochert and M. Mauve, *Implicit hop-by-hop congestion control in wireless multihop networks*, Elsevier Ad Hoc Networks **6** (2008), pp. 260–286.

A Figures (for referees)

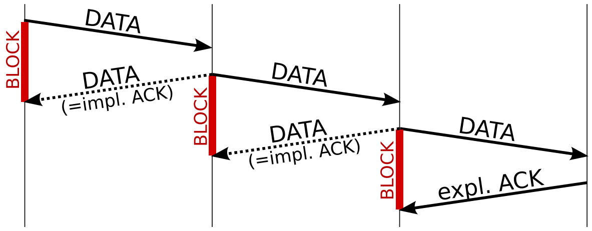
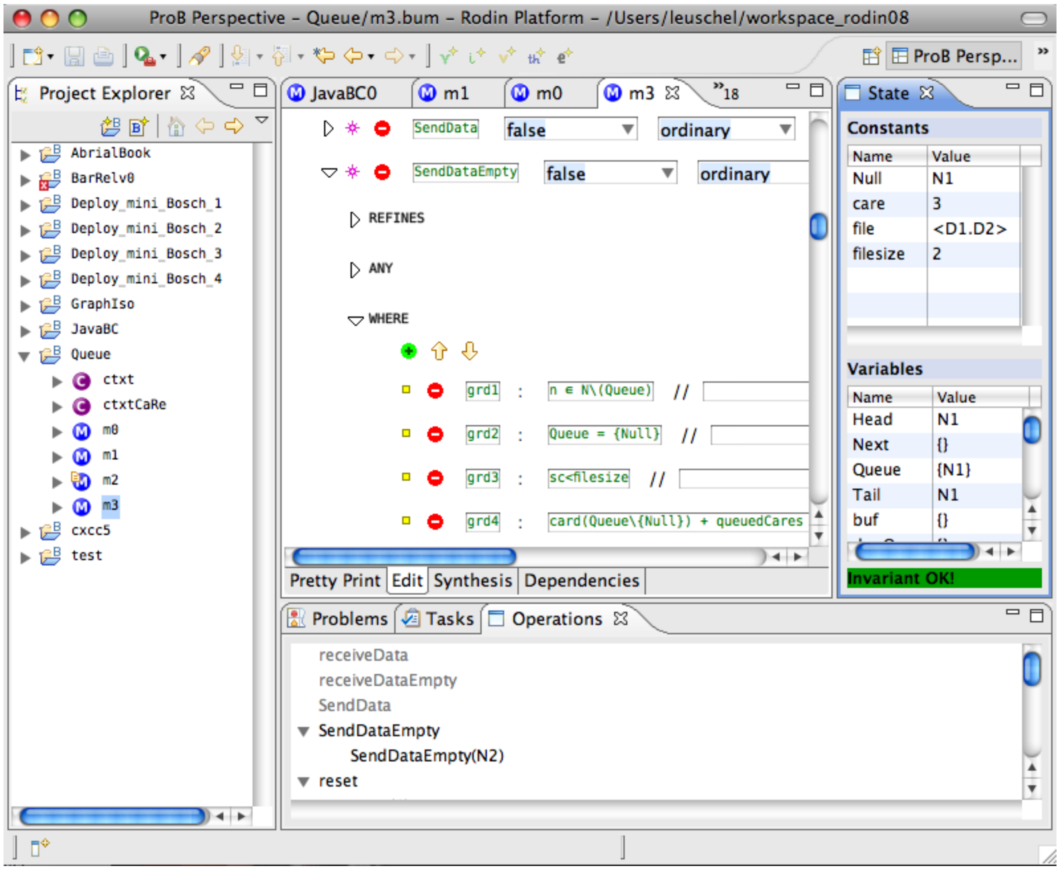


Fig. A.1. Packet forwarding in the CXCC protocol.

B Screenshots (for referees)



ProB Perspective – Queue/m3.bum – Rodin Platform – /Users/leuschel/workspace_rodin08

- AbrialBook
- BarRelv0
- Deploy_mini_Bosch_1
- Deploy_mini_Bosch_2
- Deploy_mini_Bosch_3
- Deploy_mini_Bosch_4
- GraphIso
- JavaBC
- Queue
  - ctxt
  - ctxtCaRe
  - m0
  - m1
  - m2
  - m3
- cxcc5
- test

```
▷ SendData        false  ordinary
▽ SendDataEmpty   false  ordinary
    ▷ REFINES
    ▷ ANY
    ▽ WHERE
        grd1 : n ∈ N\(Queue)  //
        grd2 : Queue = {Null}  //
        grd3 : sc<filesize  //
        grd4 : card(Queue\{Null}) + queuedCares
```

Constants

Name	Value
Null	N1
care	3
file	<D1.D2>
filesize	2

Variables

Name	Value
Head	N1
Next	{}
Queue	{N1}
Tail	N1
buf	{}

Invariant OK!

```
  receiveData
  receiveDataEmpty
  SendData
▼ SendDataEmpty
      SendDataEmpty(N2)
▼ reset
```

Fig. B.1. Third refinement animated with PROB for Rodin