Improved Probabilistic Models for 802.11 Protocol Verification

Amitabha Roy and K. Gopinath

Department of Computer Science and Automation, Indian Institute of Science, Bangalore
{aroy, gopi}@csa.iisc.ernet.in

Abstract. The IEEE 802.11 protocol is a popular standard for wireless local area networks. Its medium access control layer (MAC) is a carrier sense multiple access with collision avoidance (CSMA/CA) design and includes an exponential backoff mechanism that makes it a possible target for probabilistic model checking. In this work, we identify ways to increase the scope of application of probabilistic model checking to the 802.11 MAC. Current techniques model only specialized cases of minimum size. To work around this problem, we identify properties of the protocol that can be used to simplify the models and make verification feasible. Using these observations, we present generalized probabilistic timed automata models that are independent of the number of stations. We optimize these through a novel abstraction technique while preserving probabilistic reachability measures. We substantiate our claims of a significant reduction due to our optimization with results from using the probabilistic model checker PRISM.

1 Introduction

The IEEE 802.11 protocol [1] is a popular standard for wireless networks. Its medium access control layer (MAC) is a carrier sense multiple access with collision avoidance (CSMA/CA) design and includes an exponential backoff mechanism that makes it an ideal target for probabilistic model checking. This protocol has been modeled using a range of techniques such as finite state machines and probabilistic timed automata [2].

The 802.11 protocol suffers from a potential livelock problem, demonstrated formally in [3], which is mitigated only by the presence of a finite retry limit for each data packet. The livelock arises because it is possible, although improbable, for two stations to behave symmetrically and continuously collide until they drop their respective packets on exceeding the retry limit. In such a scenario, it is useful to bound the probability of such pathologically symmetric behavior. This motivates the application of probabilistic model checking to the problem of computing probabilities of desired and undesired behavior in the protocol. Two primary properties of interest are: the probability of the number of retries reaching a certain count and the probability of meeting a *soft* deadline.

A recent solution to the problem of obtaining these probabilities has been proposed in [2]. It models a limited (but critical) aspect of the protocol using Probabilistic Timed Automata (PTA) [4] and exploits available tools, namely, the Probabilistic Symbolic Model Checker (PRISM) [5] for computing the probability values and the real time model checker Uppaal [6] as a proof assistant. Results on the probability of the backoff counter on a station reaching a particular value and the probability of a packet being

K. Etessami and S.K. Rajamani (Eds.): CAV 2005, LNCS 3576, pp. 239–252, 2005.

transmitted within a certain deadline are presented. This work, however, models only a specialized case of two stations (sender destination pairs). When we extended the models to 3 stations (and 3 corresponding destinations), which is a practical sized network topology, we found it computationally infeasible to model check properties of interest. Also, the model has an inaccurate assumption that the packet length can vary on every retransmission.

The aim of this work is twofold. First, we present a more accurate and generalized model for the protocol that is parameterized by the number of stations. Second, we set up a logical framework to exploit protocol specific redundancies. Under this framework, we perform a number of provably correct optimizations that reduce the generalized multi station model. The optimizations involve abstracting away the deterministic waits and considering only a subset of the allowed packet sizes that nevertheless captures all the relevant behavior. In addition, we duplicate the model reduction technique of [2] for the multi station problem.

Our reduced models are immediately verifiable in PRISM and require no further tools. It is also possible to use tools like RAPTURE [7] on the reduced PTA models (see [8] for our experiences with using RAPTURE). Our results show a reduction in state space over the existing solution for two stations. We are also able to successfully model check a topology of three station that was infeasible with the current models.

The organization of the paper is as follows. We begin with the modeling formalism used in this paper. We present the generalized models for the multi station 802.11 problem and discuss the behavior of the protocol. Next, we present a notion of equivalence in probabilistic systems that abstracts away deterministic paths in the system but preserves probabilistic reachability. We give sufficient requirements for equivalence both at the level of untimed probabilistic systems and probabilistic timed automata. Based on this framework, we present our set of reductions to the generalized model for the multi station problem. We also show that we can verify soft deadlines inspite of these optimizations. We conclude with results that detail state space reduction as well as case studies for a three station topology.

2 Modeling Formalism

We need a modeling formalism that can represent the 802.11 protocol at sufficient depth and is amenable to transformations for more efficient verification. We have been guided by the existing work in [2] in our choice of Probabilistic Timed Automata to model the 802.11 protocol.

We introduce Probabilistic Timed Automata (PTA) [4], Probabilistic Systems (PS) [2, 9] and fully probabilistic systems (FPS). All these have been surveyed in [10] with special reference to their relationship in the context of probabilistic model checking.

Let χ be a set of non-negative real valued variables called clocks. Call Z the set of zones over χ, which is the set of all possible atomic constraints of the form $x \sim c$ and $(x - y) \sim c$ and their closure under conjunction. Here $x, y \in \chi$, $\sim \in \{<, \leq, >, \geq\}$ and $c \in \mathbb{N}$, where $\mathbb{N}$ is the set of natural numbers. A clock valuation v is the assignment of values in $\mathbb{R}_{\geq 0}$(where $\mathbb{R}_{\geq 0}$ is the set of non-negative reals) to all clocks in χ. The concept of a clock valuation v satisfying a zone Y, indicated as $v \triangleleft Y$, is naturally derived by

assigning values to each clock in the zone and checking whether all constraints are satisfied.

Definition 1. *A probabilistic timed automaton is a tuple* $(L, \overline{l}, \chi, \Sigma, I, P)$ *where* L *is a finite set of states,* $\overline{l}$ *is the initial state,* χ *is the set of clocks and* Σ *is a finite set of labels used to label transitions. The function* I *is a map* $I : L \rightarrow Z$ *called the invariant condition. The probabilistic edge relation* P *is defined as* $P \subseteq L \times Z \times \Sigma \times Dist(2^{\chi} \times L)$*, where* $Dist(2^{\chi} \times L)$ *is the set of all probability distributions, each elementary outcome of which corresponds to resetting some clocks to zero and moving to a state in* L*. We call a distinguished (not necessarily non-null) subset* Σ^{u} *of the set of events as* urgent events.

A critical feature of PTAs that makes them powerful modeling tools is that each transition presents *probabilistic choice* in the PTA while different outgoing probabilistic transitions from a state present *non-deterministic* choice in the PTA. Hence, a PTA can model non-determinism, which is inherent in the composition of asynchronous parallel systems.

Composition of PTAs is a cross product of states with the condition that the composed PTAs must synchronize on shared actions. For a detailed description see [2]. A feature of PTAs that is useful for higher-level modeling is urgent channels. Urgent channels are a special set of edge labels (symbols) such that time cannot be allowed to pass in a state when synchronization on an urgent channel is possible. We next define a probabilistic system (PS) (which is the same as the simple probabilistic automaton of [9]).

Definition 2. *A probabilistic system (PS), is a tuple* $(S, \overline{s}, \Sigma, Steps)$ *where* S *is the set of states,* $\overline{s}$ *is the start state,* Σ *is a finite set of labels and* $Steps$ *is a function* $Steps : S \rightarrow 2^{\Sigma \times Dist(S)}$ *where* $Dist(S)$ *is the set of all distributions over* S*.*

Definition 3. *Given a PTA* $\mathcal{T} = (L, \overline{l}, \chi, \Sigma, I, P)$*, the* semantics *of* $\mathcal{T}$ *is the PS* $[[\mathcal{T}]] = (S, \overline{s}, Act, Steps)$*, where* $S \subseteq L \times \mathbb{R}^{|\chi|}_{\geq 0}$ *is the set of states with the restrictions* $(s, v) \in S$ *iff* ($s \in L$ *and* $v \triangleleft I(s)$*) and* $\overline{s} = (\overline{l}, 0)$*.* $Act = \mathbb{R}_{>0} \cup \Sigma$*. This reflects either actions corresponding to time steps* ($\mathbb{R}_{>0}$) *or actions from the PTA* (Σ)*.* $Steps$ *is the least set of probabilistic transitions containing, for each* $(l, v) \in S$*, a set of action distribution pairs* (σ, μ) *where* $\sigma \in Act$ *and* μ *is a probability distribution over* S*.* $Steps$ *for a state* $s = (l, v)$ *is defined as follows.*
I. for each $t \in \mathbb{R}_{>0}$*,* $(t, \mu) \in Steps(s)$ *iff*

1. $\mu(l, v + t) = 1$ *and* $v + t' \triangleleft I(l)$ *for all* $0 \leq t' \leq t$*.*
2. *For every probabilistic edge of the form* $(l, g, \sigma, -) \in P$*, if* $v + t' \triangleleft g$ *for any* $0 \leq t' \leq t$*, then* σ *is non-urgent.*

II. for each $(l, g, \sigma, p) \in P$*, let* $(\sigma, \mu) \in Steps(s)$ *iff* $v \triangleleft g$ *and for each* $(l', v') \in S$*:* $\mu(l', v') = \Sigma_{X \subseteq \chi \& v' = v[X:=0]}\ p(X, l')$*, the sum being over all clock resets that result in the valuation* v'*.*

A critical result [11], analogous to the region construction result for timed automata, states that it is sufficient to assume only integer increments when all zones are closed

(there are no strict inequalities). Hence, the definition given above is modified to $S \subseteq L \times \mathbb{N}^{|\chi|}$ and $Act = \mathbb{N} \cup \Sigma$. Under integer semantics, the size of the state space is proportional to the largest constant used. For the rest of this paper, we will assume integer semantics. Note that, in the presence of non-determinism, the probability measure of a path in a PS is undefined. Hence, define an adversary or scheduler that resolves non-determinism as follows:

Definition 4. *An adversary of the PS* $\mathcal{P} = (S, \overline{s}, Act, Steps)$ *is a function* $f : S \rightarrow \cup_{s \in S} Steps(s)$ *where* $f(s) \in Steps(s)$.

We only consider *simple* adversaries that do not change their decision about an outgoing distribution every time a state is revisited, their sufficiency has been shown in [12]. A simple adversary induces a Fully Probabilistic System (FPS) as defined below.

Definition 5. *A simple adversary* A *of a PS* $\mathcal{P} = (S, \overline{s}, Act, Steps)$ *induces an FPS or Discrete Time Markov Chain* $\mathcal{P}^A = (S, \overline{s}, P)$. *Here,* $P(s) = A(s)$, *the unique outgoing probability distribution for each* $s \in S$, *where we drop the edge label on the transition.*

Given a PS $\mathcal{M}$ and a set of "target states" F, consider an adversary A and the corresponding FPS $\mathcal{M}^A$. A probability space ($Prob^A$) may be defined on $\mathcal{M}^A$ via a cylinder construction [13]. A path ω in $\mathcal{M}^A$ is simply a (possibly infinite) sequence of states $\overline{s}s_1s_2...$ such that there is a transition of non-zero probability between any two consecutive states in the path. For model checking, we are interested in $ProbReach^A(F) \stackrel{def}{=} Prob^A\{\omega \in Path^A_\infty \mid \exists i \in \mathbb{N} \text{ where } \omega(i) \in F\}$. F is the desired set of target states, $\omega(i)$ is the i^{th} state in the path ω and $Path^A_\infty$ represents all infinite paths in $\mathcal{M}^A$. Define $MaxProbReach^M(F)$ and $MinProbReach^M(F)$ as the supremum and infimum respectively of $\{ProbReach^A(F)\}$ where the quantification is over all adversaries. This definition does not take into account sink states with no outgoing transitions. However, these states can easily be handled by adding self loops.

Properties of interest at the PTA level are specified using Probabilistic Computational Tree Logic (PCTL) formulas [14]. We limit ourselves to restricted syntax (but non trivial) PCTL formulas, expressible as $P_{\sim\lambda}\{\Diamond p\}$, where $\sim \in \{<, >, \leq, \geq\}$, λ is the constant probability bound that is being model checked for and p is a proposition defined for every state in the state space. These PCTL formulas translate directly into a probabilistic reachability problem on the semantic PS corresponding to the PTA. The reason for this restriction is that, in the case of the 802.11 protocol, the properties of interest, including the real time ones, are all expressible in this form. In this restricted form of PCTL, we indicate numerical equivalence using the following notation.

Definition 6. *Two PSs* $\mathcal{P}_1$ *and* $\mathcal{P}_2$ *are equivalent under probabilistic reachability of their respective target states* F_1 *and* F_2, *denoted by* $\mathcal{P}_1 \stackrel{PS}{\equiv}_{F_1,F_2} \mathcal{P}_2$ *when* $MaxProbReach^{\mathcal{P}_1}(F_1) = MaxProbReach^{\mathcal{P}_2}(F_2)$ *and* $MinProbReach^{\mathcal{P}_1}(F_1) = MinProbReach^{\mathcal{P}_2}(F_2)$.

Definition 7. $PTA_1 \stackrel{PTA}{\equiv}_{\phi_1,\phi_2} PTA_2$ *when* $[[PTA_1]] \stackrel{PS}{\equiv}_{F_1,F_2} [[PTA_2]]$. *The criterion for marking target states is that* F_1 *corresponds to the target states in the reachability problem for the PCTL formula* ϕ_1, *while* F_2 *corresponds to the target states for the PCTL formula* ϕ_2.

3 Probabilistic Models of 802.11 Protocol

In this section, we present generalized probabilistic models of the 802.11 basic access MAC protocol assuming no hidden nodes[1]. The model for the *multi-station* 802.11 problem consists of the station model and a shared channel, shown in Figures 2 part (a) and 1 part(b) respectively. We assume familiarity with conventions used in graphical representation of timed automata. The states marked with a 'u' are urgent states while that marked by concentric circles is the start state. The station models are replicated to represent multiple sender-destination pairs. Some critical state variables are: bc that holds the current backoff counter value, tx_len that holds the chosen transmission length and *backoff* that represents the current remaining time in backoff. The function $RANDOM(bc)$ is a modeling abstraction that assigns a random number in the current contention window. Similarly, $NON\_DET(TX\_MIN, TX\_MAX)$ assigns a non-deterministic packet length between TX_MIN and TX_MAX, which are the minimum and maximum allowable packet transmission times respectively. The values used for verification are from the Frequency Hopping Spread Spectrum (FHSS) physical layer [1]. The transmission rate for the data payload is 2 Mbps.

The station automaton shown in Figure 2, begins with a data packet whose transmission time it selects non-deterministically in the range from $258\mu s$ to $15750\mu s$. On sensing the channel free for a Distributed InterFrame Space ($DIFS = 128\mu s$), it enters the $Vulnerable$ state, where it switches its transceiver to transmit mode and begins transmitting the signal. The $Vulnerable$ state also accounts for propagation delay. It moves to the $Transmit$ state after a time $VULN = 48\mu s$ with a synchronization on $send$. After completing transmission, the station moves to $Test\_channel$ via one of the two synchronizations, $finish\_correct$ on a successful transmission and $finish\_garbled$ on an unsuccessful transmission. The channel keeps track of the status of transmissions, going into a garbled state whenever more than one transmission occurs simultaneously. The station incorporates the behavior of the destination and diverges depending on whether the transmission was successful, or not. If the transmission was successful, the portion of the station corresponding to the destination waits for a Short InterFrame Space ($SIFS = 28\mu s$) before transmitting an ack, which takes $ACK = 183\mu s$.

On an unsuccessful transmission, the station waits for the acknowledgment timeout of $ACK\_TO = 300\mu s$. It then enters a backoff phase, where it probabilistically selects a random backoff period *backoff*$= RANDOM(bc)$, with uniform probability, a value from the contention window (CW) given by the range $[0, (C + 1).2^{bc} - 1]$, where C is the minimum CW ($15\mu s$ for the FHSS physical layer). The backoff counter (bc) is incremented each time the station enters backoff. The backoff counter is frozen when a station detects a transmission on the medium while in backoff.

It may be noted that the channel model in [2] is aware of exactly which stations are transmitting; for n stations, there are 2^n possibilities leading to the channel having $\Omega(2^n)$ state space. Our design recognizes the fact that it is sufficient for the channel to be aware of the number of transmitters using the tx_count variable. Hence our channel model has atmost a constant number of states plus a linear factor in terms of n leading to $O(n)$ states.

[1] In the absence of hidden nodes [15], the channel is a shared medium visible to all the stations.

Also, we start with an abstracted station model, which incorporates the deterministic destination. The validity of this abstraction for the two station case has been shown in [2]. The extension to the multi station case is given in [8].

4 Reducing State Space by Compression of Deterministic Paths

In the 802.11 protocol, there are numerous cases where the component automata representing the system simply count time or where different resolutions of non-determinism lead to the same state but through different paths. If we are verifying an untimed property then such execution fragments increase state space without any contribution to probabilistic reachability. We discovered on studying these models that it is possible to derive alternative *optimized* probabilistic timed automata that avoid the cost of such unnecessary deterministic behavior by compressing these deterministic paths into equivalent but shorter paths. The problem is the lack of a suitable formalism to support our optimizations. This section provides a framework that can be used to justify the equivalence of our optimized models to the original ones.

We assume that the state space is a subset of an implicit global set of states. This allows operations such as intersection and union between the set of states of two different automata. In particular, for this paper we consistently name states across the automata we consider. Our objective is to formalize "deterministic" behavior of interest. The key relationship used in this formalization is a specialization of dominators as defined in [7]. We refer to this restricted version of dominators as "deterministic dominators" in the rest of this paper.

Definition 8. *For a distribution π over the* finite *elementary event set X, define the support of the distribution as $supp(\pi) = \{x \in X \mid \pi(x) > 0\}$*

Definition 9. *Given a PS consisting of the set of states S, define $\prec_D$ as the smallest relation in $S \times S$ satisfying the following: $\forall s \in S \; s \prec_D s$ and $\forall t \in S \; [\forall (a, \pi) \in Steps(s) : \exists x \;\; (supp(\pi) = \{x\}) \wedge (x \prec_D t) \Rightarrow s \prec_D t]$*

If the relation $s \prec_D t$ holds then we say that t is the deterministic dominator of s.

An example of a deterministic dominator is shown in the PSs of Figure 1 part(a), where $S \prec_D T$.

Definition 10. *Given distributions P_1 over S_1 and P_2 over S_2, define $P_1 \stackrel{dist}{\equiv} P_2$ when $supp(P_1) = supp(P_2) = S$ and $\forall s \in S$ we have $P_1(s) = P_2(s)$.*

Based on the notion of equivalence of distributions, we define the notion of equivalence of *sets* of distributions. Let $Steps_1$ be a set of labeled distributions over S_1 and $Steps_2$ be a set of labeled distributions over S_2.

Definition 11. *$Steps_1 \stackrel{dist}{\equiv} Steps_2$ whenever $\forall (a, \mu_1) \in Steps_1 \; \exists (b, \mu_2) \in Steps_2$ such that $\mu_1 \stackrel{dist}{\equiv} \mu_2$ and $\forall (a, \mu_2) \in Steps_2 \; \exists (b, \mu_1) \in Steps_1$ with $\mu_2 \stackrel{dist}{\equiv} \mu_1$.*

Definition 12. *A path in the PS $\mathcal{P} = (S, \overline{s}, \Sigma, Steps)$ is a sequence of state-action pairs $(s_1, a_1), (s_2, a_2)..(s_{n+1})$ such that $\forall i \in \{1..n\}$ we have $\exists (a_i, \mu) \in Steps(s_i)$ such that $\mu(s_{i+1}) > 0$.*

4.1 Deterministic Path Compression in Probabilistic Systems

Consider the two PSs of Figure 1 part(a), each of which has the start state U. It should be clear that each of $MaxProbReach(X)$ and $MinProbReach(X)$ takes the same value in both the systems since we have only removed (compressed) the deterministic segment $B \rightarrow C$.

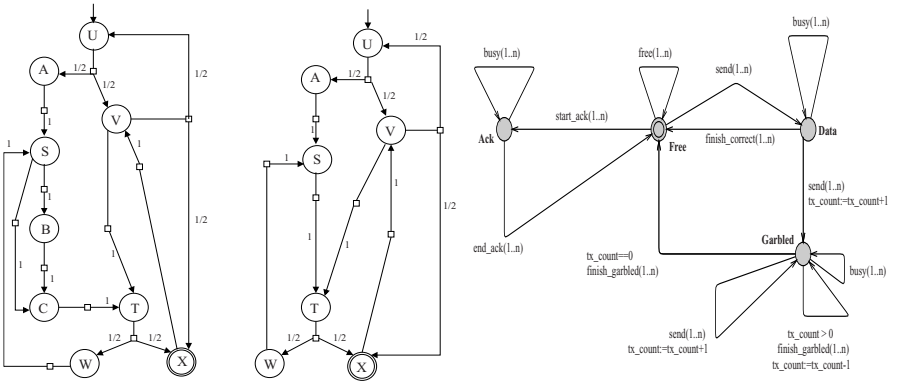

Fig. 1. (a)Two Related Probabilistic Systems (left); (b)PTA model for the Channel - Generalized for the multiple station case(right)

We formalize this notion of deterministic path compression at the level of PSs in theorem 1.

Consider two *finite* Probabilistic Systems $PS_1 = (S_1, \overline{s}, Act, Steps_1)$ and $PS_2 = (S_2, \overline{s}, Act, Steps_2)$ with an identical set of actions. All transitions in $Steps_1$ and $Steps_2$ are simple transitions of the form (s, a, μ) where s is the originating state, $a \in Act$ and μ is a probability distribution over the state space. Note that the S_1 and S_2 are necessarily not disjoint because of the common start state $\overline{s}$.

Definition 13. *If, for some $s \in S_1 \cap S_2$, $Steps_1(s) \stackrel{dist}{\equiv} Steps_2(s)$ does not hold then s is a point of disagreement between the two PSs.*

Theorem 1 (Equivalence in PSs). *Given two PSs $PS_1(S_1, \overline{s}, Act, Steps_1)$ and $PS_2(S_2, \overline{s}, Act, Steps_2)$ satisfying the following conditions:*

1. *For any state $s \in S_1 \cap S_2$, if s is a point of disagreement then $\exists t \in S_1 \cap S_2$ such that, t is not a point of disagreement and in each of the systems, $s \prec_D t$.*
2. *Let $F_1 \subseteq S_1$ and $F_2 \subseteq S_2$ be sets of target states we are model checking for. We impose the condition $S_1 \cap S_2 \cap F_1 = S_1 \cap S_2 \cap F_2$. For every $s \in S_1 \cap S_2$, which is a point of disagreement we have the following: For the postulated deterministic dominator t and for every state u on any path in PS_1 between s and t, $u \in F_1 \Rightarrow (s \in F_1) \vee (t \in F_1)$. Similarly, for every state u on any path in PS_2 between s and t, $u \in F_2 \Rightarrow (s \in F_2) \vee (t \in F_2)$.*

Under these conditions, $PS_1 \stackrel{PS}{\equiv}_{F_1,F_2} PS_2$.

The proof follows from first principles by setting up a bijective mapping between paths in the two PSs. The complete proof is available in [8].

4.2 A Comparison Framework for PTAs

Given PTA_1 and PTA_2 and their respective restricted PCTL requirements ϕ_1 and ϕ_2, we need a set of conditions under which we may claim $PTA_1 \overset{PTA}{\equiv}_{\phi_1,\phi_2} PTA_2$. By Definition 7, this is equivalent to showing that $[[PTA_1]] \overset{PS}{\equiv}_{F_1,F_2} [[PTA_2]]$, where F_1 and F_2 are the corresponding target states of ϕ_1 and ϕ_2 respectively. Our optimizations are based on deterministic path compression as outlined in Section 4.1. Hence, we impose requirements on PTA_1 and PTA_2 under which we can apply theorem 1 to $[[PTA_1]]$ and $[[PTA_2]]$ to deduce $[[PTA_1]] \overset{PS}{\equiv}_{F_1,F_2} [[PTA_2]]$.

Consider two PTAs with an identical set of clocks and events: $PTA_1 = (L_1, \overline{l_1}, \chi, \Sigma, I_1, P_1)$ and $PTA_2 = (L_2, \overline{l_2}, \chi, \Sigma, I_2, P_2)$. We assume that the automata have the same set of urgent events, Σ^u.

Definition 14. *A state $s \in L_1 \cap L_2$ is a point of disagreement between the two probabilistic timed automata if either they differ on the invariant or they differ in the set of outgoing transitions or both. Taking a transition out of a state s as the tuple $(s, z, \sigma, P(2^\chi \times L))$, call two transitions different if they disagree on either the guard z, or the event label on the transition σ, or the distribution $P(2^\chi \times L)$.*

The semantic PSs are $[[PTA_1]]$ and $[[PTA_2]]$ respectively. Let $States([[PTA_1]])$ and $States([[PTA_2]])$ denote states of the semantic PSs for PTA_1 and PTA_2 respectively. The states in the semantic PS are tuples (s, v) where s is a state of the PTA and v is a clock valuation.

Lemma 1. *A state $(s, v) \in States([[PTA_1]]) \cap States([[PTA_2]])$ as a point of disagreement (with regard to condition 1 of theorem 1) between the two PS implies that s is a point of disagreement between PTA_1 and PTA_2.*

The condition that labels should also be identical might seem too restrictive considering that we are only interested in probabilistic reachability. However, the next set of lemmas will show that when composing PTAs labels are important.

Most real world systems and the 802.11 protocol in particular are modeled as a composition of PTAs. In a composed system, the above lemma will only tell us whether a particular common state in the PTA can generate a point of disagreement in the semantic PS. This common state represents the composed state of all the PTAs composing the model. The next few lemmas extend lemma 1 to the scenario of composed probabilistic timed automata.

Definition 15. *Consider two PTAs formed of compositions, as follows.*
$PTA_1 = PTA_1^1 \parallel PTA_2^1 \parallel PTA_3^1 \parallel .. \parallel PTA_n^1$ *and*
$PTA_2 = PTA_1^2 \parallel PTA_2^2 \parallel PTA_3^2 \parallel .. \parallel PTA_n^2$.
Define the difference set as the set $D \subseteq \{1, 2, .., n\}$ such that $\forall i \in D : PTA_i^1 \neq PTA_i^2$ and $\forall i \notin D : PTA_i^1 = PTA_i^2$. By equality we mean exactly the same automaton in both the compositions (component wise equality of the tuples defining them).

Definition 16. *We define the specific difference set for the index $i \in D$ as $D_i \subseteq states(PTA_i^1) \cap states(PTA_i^2)$ where D_i is the set of states that disagree across the automata as outlined in definition 14. For every $i \notin D$ set $D_i = \emptyset$.*

Lemma 2. *Consider the composed PTA models of Definition 15. Let S_{common} be the set of common states between PTA_1 and PTA_2. A composed state in S_{common}, say $(l_1, l_2, .., l_n)$ is a point of disagreement between PTA_1 and PTA_2 implies that at least one automaton is in its specific difference set.*

In the composed PTAs of definition 15, each state in the semantic PS for a PTA is a combination of states and clock valuations of the individual PTA in the composition. The next lemma combines lemmas 1 and 2.

Lemma 3 (PTA level requirements).
A state in $States([[PTA_1]]) \cap States([[PTA_2]])= (l_1, l_2.., l_n, v)$ as a point of disagreement implies that for at least one $i \in \{1..n\}$, the common state l_i of both PTA_i^1 and PTA_i^2 is an element of their specific disagreement set.

Lemma 3 identifies precisely those states in the *component* PTA that *may* cause a disagreement in the PS for the composed system.

4.3 Proof Technique

We will use the framework in this section to prove the correctness of our reduced models. Although our objective is the 802.11 protocol, the concept of deterministic path compression has been developed in a generalized manner anticipating its application to other protocols.

To prove that a reduced PTA model (PTA_2) corresponding to the original PTA model (PTA_1) is correct, we need to prove that $PTA_1 \overset{PTA}{\equiv}_{\phi_1,\phi_2} PTA_2$. Here ϕ_1 and ϕ_2 are the corresponding PCTL formulas in the two models. For our purposes $\phi_1 = \phi_2$ since we are interested in proving that we will arrive at the same result for the same particular PCTL formula. We proceed with the proof in the following manner.

1. Identify the difference set (Definition 15). Compute the specific difference set of each component automaton in the difference set using Definition 16. This is easily done by a visual inspection of the automata.
2. Identify composed states where one or more automata are in their specific difference set. At this point we use protocol specific proofs to limit such combinations to a manageable size. From Lemma 2 we know the set of composed states obtained in this step is a superset of the actual difference set across the composed PTA.
3. For each composed state, we argue about the possible evolution of the untimed model obtained through Definition 3. We show that
i) There is the same deterministic dominator in each of $[[PTA_1]]$ and $[[PTA_2]]$. This is usually the hardest part of the proof. However, we use the fact that the deterministic dominator state in the PS is expressible as the combination of a composed state and clock valuation in the PTA. Hence the proofs are in terms of the PTA rather than the PS. We generally show that each component automaton reaches the state in the composition and progress can only be made when the entire model is in the composed state.

ii) Final states in $[[PTA_1]]$ and $[[PTA_2]]$, corresponding to the PCTL formulas ϕ_1 and ϕ_2 respectively, are distributed as specified in condition 2 of Theorem 1.
iii) PTA_1 and PTA_2 have the same start state.

From Lemma 3 we know that this is sufficient for Theorem 1 to hold. Hence we conclude that at the level of PTAs $PTA_1 \overset{PTA}{\equiv}_{\phi_1,\phi_2} PTA_2$.

Deterministic Path Compression, at the level of PSs, bears similarity to weak bisimulation [9] that can abstract away internal actions. However, a notable difference in our approach from weak bisimulation is that we are able to change invariants on states in the PTA. This corresponds to removing time steps (Definition 3) in the corresponding semantic PS. These time steps are *not* internal actions because composed PSs must synchronize on time steps to maintain the semantics of PTA composition. A possibility would be to apply weak bisimulation to the final composed model but this would mean fixing the number of stations in the composition. The reduced models would no longer be valid for the general multi station problem.

5 Reducing the 802.11 Station Automaton

For the 802.11 problem, we optimize the station automaton in multiple steps, starting from the original abstract station model of Figure 2 part (a). In each case, the set of final states correspond to the PCTL formula $\phi = P_{<\lambda}[\diamond(bc = k)]$, which expresses the property that the backoff counter of some station reaches k. For every reduction from PTA_1 to PTA_2, we prove the correctness of our optimizations by showing that $PTA_1 \overset{PTA}{\equiv}_{\phi,\phi} PTA_2$. Due to space constraints, we defer complete proofs to [8] and only motivate the key ideas. Our proofs are driven by behavior exhibited by the 802.11 PTA models. For example, a key aspect of many of our proofs is the fact that 802.11 backoff counters are frozen when a busy channel is detected. We can essentially ignore stations in backoff when the channel is busy. These proofs have been constructed to be independent of the number of stations in the composition.

Our first optimization removes the $SIFS$ wait following a successful transmission. The original model is $AbsLAN = AbsStn_1 \parallel AbsStn_2 \parallel .. \parallel AbsStn_n \parallel Chan$ and the reduced model is $IntLAN = IntStn_1 \parallel IntStn_2 \parallel .. \parallel IntStn_n \parallel Chan$. The intermediate station model $IntStn$ has the $SIFS$ wait removed and is shown in Figure 2 part (b). The difference set (see Definition 15) includes all the stations and does not include the channel, which is unchanged. The specific difference set is only the $Test\_Channel$ urgent state immediately after asserting $finish\_correct$. The key idea of the proof is as follows: All the other stations will detect the busy channel and move into the $Wait\_until\_free$ or $Wait\_until\_free\_II$ state. The successfully completing station will move into the $Done$ state while the rest of the stations will move either into $Wait\_for\_DIFS$ or $Wait\_for\_DIFS\_II$ states, which gives us a deterministic dominator in both the automata ($AbsLAN$ and $IntLAN$). In the proof, we exploit the fact that in the 802.11 protocol, the backoff counters are frozen when a transmission is detected on the channel. This is modeled by the station in *Backoff* moving into the $Wait\_until\_free\_II$ state.

In the final reduced station model, used in our experiments, the $DIFS$ wait has also been removed. Proving the deterministic dominator relationship is a little more com-

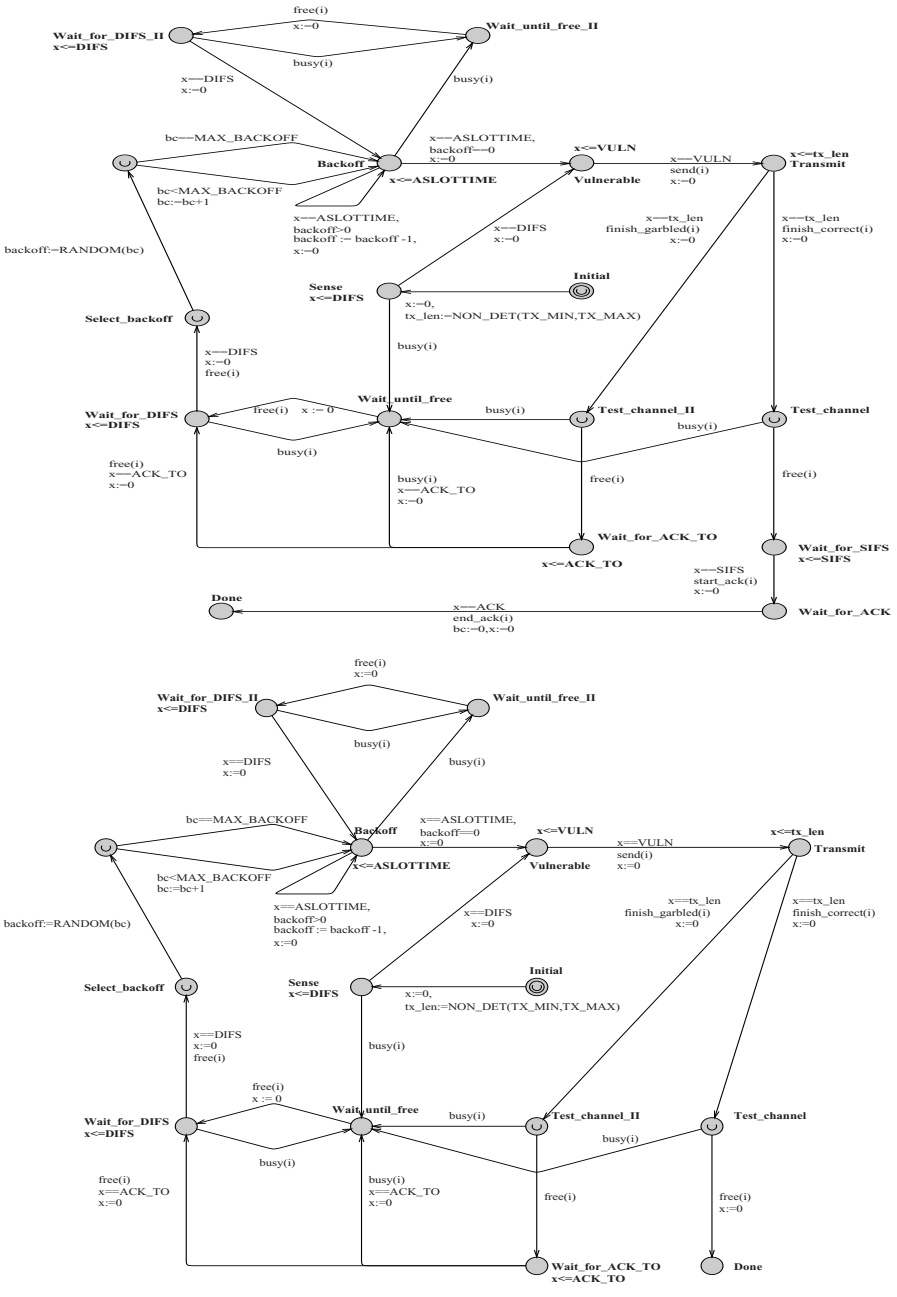

Fig. 2. (a) PTA model for an Abstract Station representing both the sender and destination (top); (b) PTA model for an Intermediate Abstracted and Reduced Station - ACK protocol removed (bottom)

plicated here because we need to consider both collision and successful transmission cases. A discussion of the steps involved can be found in [8].

The major contributor of state space in the protocol is the large range of allowed transmission lengths. The range is from $315\mu s$ to $15717\mu s$ and this proves to be a significant impediment.

To overcome this problem, we begin by parameterizing our models as follows. Rather than having a non-deterministic edge that selects packet lengths, which are subsequently held constant, we *parameterize* the models by packet length and remove the non-deterministic choice. Hence, we now have a *series* of PTA models depending on the choice of parameterizations. The allowable assignment of packet (transmission) lengths is from Par^{full}, the set of all possible parameterizations. Each of $tx\_len_1, .., tx\_len_n$ is assigned a value from the interval $[TX\_MIN, TX\_MAX]$. Formally, $Par^{full} = [TX\_MIN, TX\_MAX]^n$.

Consider the reduced set of parameterizations $Par^{reduced} \subset Par^{full}$ where $tx\_len_1 = TX\_MIN$ and $tx\_len_{i+1} - tx\_len_i \leq VULN$, $1 \leq i < n$. Here we restrict the maximum allowable increase in transmission length of one station over its immediate predecessor. This eliminates many parameterizations that would have assigned transmission lengths close to maximum resulting in a large state space. We have shown (see [8] for details) that it is sufficient to consider only this limited range of transmission lengths.

6 Soft Deadline Verification

The probability of meeting soft deadlines, which is the minimum probability of a station delivering a packet within a certain deadline, is a real time property that can be formulated as a probabilistic reachability problem. For example, in an 802.11 topology of three senders and three receivers, we are interested in the probability that *every* station successfully transmits its packet within a given deadline. The reductions presented in this paper, which depend on deterministic path compression, do not preserve total time elapsed since certain states in the probabilistic timed automata where the composite model can spend time have been removed. As a result, paths are replaced with shorter (time wise) versions.

However, one key aspect of our reductions is that they affect deterministic and well-defined segments of the automata. The intuition is that it should be possible to "compensate" for the reductions by using additional available information. For example, removing the acknowledgment protocol has the effect of subtracting a $SIFS + ACK$ period for every successful transmission made. On the other hand removing $DIFS$ wait results in subtracting $DIFS$ from the elapsed time for any transmission made.

We begin with the traditional "decoration" of a PTA in order to verify real time properties. Assume the existence of a composed state $Done$, which is the composition of the state $Done$ across the components the model. Decorating the PTA involves adding a global clock (say y) to the system that counts total time elapsed and a state $Deadline\_exceeded$. Edges are added from each state other than $Done$, with guard $y \geq deadline$ to $Deadline\_exceeded$. Every invariant other than at $Done$ and $Deadline\_exceeded$ is taken in conjunction with $y \leq deadline$. The objective is to model check for the PCTL formula $P_{>\lambda}[\Diamond Done]$, which expresses the soft deadline property. We defer further discussion of the details to [8] due to lack of space.

7 Results

Our verification platform is a 1.2 GHz Pentium III server with 1.5 GB of ECC memory and running Linux 2.4. Our experiments used the Multi-Terminal Binary Decision Diagram (MTBDD) engine of PRISM. All properties were checked with an accuracy of 10^{-6}, which means that the model checker stops when probabilities returned by successive iterations differ by, or less than, this value.

The growth in state space for the multi station problem is shown in Table 2 part (a). The optimized two station models show a significant improvement in size when compared with the models of [2]. Unoptimized models for three and four stations cannot even be built by the model checker within the resources provided. The obtained upper bounds on the probability of the backoff counter reaching a certain value are shown in Table 1. The values for a three station model are higher due to increased contention for the channel. Verification costs for our optimized models are clearly lower.

Table 1. Probability of backoff counter reaching a specified value in 2 station and 3 station cases

Backoff Counter	2original (secs)	2optimized (secs)	Maximum probability	3optimized Iterations	3opt (secs)	Maximum probability
1	0.69	0.09	1.0	285	1428	1.0
2	8.95	1.15	0.18359375	107	124	0.59643554
3	37.37	6.29	0.0170326	259	1250	0.104351032
4	113.25	29.12	7.9424586e-4	506	14183	0.008170952
5	327.04	120.5	1.8566660e-5	525	37659	2.83169319e-4
6	970.38	508.26	2.1729427e-7	947	246874	2.85355921e-5

Table 2. State space sizes for the backoff counter problem and soft deadline problem results

Stations	2Orig	2Opt	3	4
States	5958233	393958	1084111823	1377418222475
Transitions	16563234	958378	3190610466	5162674182210
Choices	11437956	598412	1908688031	2958322202754

(a) State space size

G.729 type	Time (sec)	Min Probability
1	613	0
2	52388	0.0117

(b) Soft deadline results

We include results from an example case study involving soft deadlines. Consider three overlapping 802.11 wireless networks each servicing seven 802.11 stations. Assume voice data being distributed to all stations from a 100 Mbps 802.3 LAN through the wireless network using either of two subtypes of the G.729 [16] voice encoding scheme. A soft deadline for meeting the resultant bandwidth constraints can be formulated; for details see [8]. The probability of meeting this deadline is shown in Table 2 part (b).

8 Conclusion

In this paper, we have introduced generalized probabilistic timed automata models for the 802.11 MAC and optimized them using deterministic path compression, a novel

technique to remove protocol redundancies. We have been somewhat successful, using this optimization in tackling the state space problem for the 802.11 wireless LAN protocol. We have also shown that it is still possible to compute the minimum probability of meeting soft deadlines with the optimized models.

Future extenstions to this effort are to model check four or more stations as well as consider extensions to the basic access protocol considered here.

References

1. The Institute of Electrical and Inc. Electronics Engineers. *IEEE Std. 802.11 - Wireless LAN Medium Access Control(MAC) and Physical Layer (PHY) specifications*, 1999.
2. Marta Kwiatkowska, Gethin Norman, and Jeremy Spronston. Probabilistic model checking of the IEEE 802.11 wireless local area network protocol. In *Proc. PAPM/PROBMIV'02*, volume 2399, pages 169–187. Springer, LNCS, 2002.
3. Moustafa Youssef, Arunchandar Vasan, and Raymond Miller. Specification and analysis of the dcf and pcf protocols in the 802.11 standard using systems of communicating machines. In *IEEE ICNP 2002*, November 2002.
4. Marta Kwiatkowska, Gethin Norman, Roberto Segala, and Jeremy Sproston. Automatic verification of real-time systems with discrete probability distributions. *Lecture Notes in Computer Science*, 1601:75–95, 1999.
5. M. Kwiatkowska, G. Norman, and D. Parker. PRISM: Probabilistic symbolic model checker. In T. Field, P. Harrison, J. Bradley, and U. Harder, editors, *Proc. 12th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS'02)*, volume 2324 of *LNCS*, pages 200–204. Springer, 2002.
6. Kim Guldstrand Larsen, Paul Pettersson, and Wang Yi. UPPAAL in a nutshell. In *International Journal on Software Tools for Technology Transfer*, volume 1, pages 134–152, 1997.
7. P.R. D'Argenio, Bertrand Jeannet, Henrik E. Jensen, and Kim G. Larsen. Reduction and refinement strategies for probabilistic analysis. In *Process Algebra and Probabilistic Methods. Performance Modeling and Verification : Second Joint International Workshop PAPM-PROBMIV 2002*, volume 2399. LNCS, 2002.
8. http://agni.csa.iisc.ernet.in/∼gopi/aroy/paper.pdf.
9. Roberto Segala and Nancy Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.
10. Marta Kwiatkowska. Model checking for probability and time:from theory to practice invited paper. In *Proc.* 18^{th} *IEEE Symposium on Logic in Computer Science (LICS'03)*, pages 351–360. IEEE Computer Society Press, 2003.
11. Marta Kwiatkowska, Gethin Norman, and Jeremy Sproston. Probabilistic model checking of the 802.11 wireless local area network protocol. Technical Report CSR-02-05, School of Computer Science,University of Birmingham, 2002.
12. Christel Baier and Marta Z. Kwiatkowska. Model checking for a probabilistic branching time logic with fairness. *Distributed Computing*, 11(3):125–155, 1998.
13. John G. Kemeney, J. Laurie Snell, and Anthony W. Knapp. *Denumerable Markov Chains*. Springer Verlag, 1976.
14. Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.
15. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Comput. Networks*, 38(4):393–422, 2002.
16. International Telecommunication Union. *Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP)*, 1996.