Verifying Soft Deadlines with Probabilistic Timed Automata*

Marta Kwiatkowska[1], Gethin Norman[1], Roberto Segala[2] and Jeremy Sproston[1]

[1] University of Birmingham, Birmingham B15 2TT, United Kingdom
{M.Z.Kwiatkowska,G.Norman,J.Sproston}@cs.bham.ac.uk
[2] Dipartimento di Scienze dell'Informazione, Università di Bologn a,
Mura Anteo Zamboni 7, 40127 Bologna, Italy
segala@cs.unibo.it

Abstract. This paper describes work in progess performed as part of an ongoing project aimed at the development of theoretical foundations and model checking algorithms for the verification of *soft deadlines* in timed systems, that is, properties such as "there is a 90% chance that the message will be delivered within 5 time units". The research is focussed on the probabilistic timed automata model [11], an extension of timed automata [3], and includes: model checking of discrete-probabilistic automata based on the region graph construction [11]; symbolic methods based on forwards and backwards reachability [10,13]; and the continuous probabilistic timed automata [12].

1 Introduction

The design and analysis of many systems, to mention communication protocols, embedded systems and multimedia protocols, requires detailed knowledge of their *real-time* aspects, in addition to the functional requirements. Recent advances in software technology underpinning verification tools have brought *automatic verification* of real-time systems into the realm of industrial applications. This is evident through the success of case studies, ranging from the gear box controller to Philips audio protocol, performed with the help of model checkers such as Uppaal.

However, existing tools can only verify deterministic (*hard*) deadlines, i.e. properties such as "if the packet is sent then it will be delivered *within* 80*ms*". In the presence of lossy media or faulty hardware, hard deadlines can be too restrictive. *Probabilistic* (*soft*) deadlines provide a viable alternative; these express *the probability* of a certain target of *quality of service* being achieved (here delivery occurring within 80*ms with probability at least 90%/at most 3%*). For applications such as audio or multimedia protocols, which process and transmit *continuous* media data, it is often necessary to allow *stochastic timing*, in the sense that the user could specify that packets are sent according to exponential

* Supported in part by EPSRC grants GR/M04617, GR/M13046 and GR/N22960.

or normal distribution. In such cases, measures such as mean time to delivery are additionally required.

This paper describes work in progress performed as part of an ongoing research project. The project is centered upon the probabilistic timed automata model [11], an extension of the timed automata [3], with the help of which soft deadlines can be specified and verified. The aims of the project are to derive efficient model checking algorithms and industrially-relevant case studies for the verification of soft deadlines.

2 Background

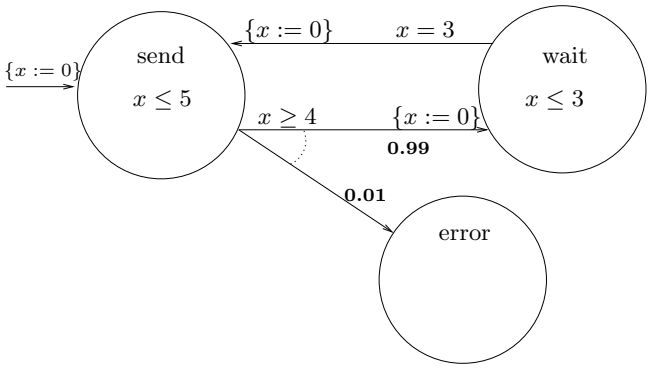
The design and analysis of many hardware and software systems, for example embedded systems, monitoring equipment, communication and multimedia protocols, requires detailed knowledge of their real-time aspects, in addition to the functional requirements. Typically, this is expressed in terms of *real-time constraints*, for example, "video frames arrive at the display device within a time bound of 50 to 100 milliseconds after being sent". In the case of safety-critical systems, such as hospital monitoring equipment, vehicle controllers, etc., it is essential to ensure that such constraints are never invalidated. However, in many other cases, for example audio and multimedia protocols, such *hard deadlines* are too strict: a satisfactory approach is to determine that the packets arrive *mostly* within the specified time bounds. This reflects the fact that violating a hard deadline does not affect the functionality of the protocol, but only its *quality of service*; for example, the audio and video streams may be misaligned in not more than 4% of the cases.

The term 'quality of service' originates from multimedia systems and refers to *quantitative* estimates of the percentage/probability of some target (e.g. delivery of a packet within a time bound) of quality of service being satisfied. Hard deadlines are known as deterministic, and are supplemented with so-called *soft deadlines* where appropriate (these are also known as *probabilistic deadlines* [1]). An example of a soft deadline is the statement "*with probability at least* 0.9, video frames arrive at the display device within a time bound of 50 to 100 milliseconds after being sent". Soft deadlines are also useful when analysing the behaviour of systems in presence of lossy channels or faulty hardware: they help specify *fault-tolerance* and *reliability* properties such as "deadlock will not occur with probability 1", or "message may be lost with probability at most 0.01". A more complex scenario arises when it is necessary to consider *stochastic timing*, that is, soft deadlines must be derived under the assumption that some set of events is governed by a certain *continuous time* probability distribution. For example, the user could specify that the rate of arrivals of video frames is normal with mean of 40 ms and variance of 5 ms, and service is exponential with rate 45 ms. In such cases, estimating *performance characteristics* such as throughput and mean service time is desirable, in addition to soft deadlines.

Formal methods and notations are invaluable when analysing real-time aspects of systems. The *timed automata* model of Alur and Dill [3] has proved very

popular. A timed automaton is an ordinary automaton extended with real-valued *clocks* which increase uniformly with time. Clocks may be compared to integer time bounds to form clock constraints such as $(x \geq 5) \wedge (x \leq 10)$. There are two types of clock constraints: *invariants* labelling nodes, and *guards* labelling transitions. The automaton may only stay in a node, letting time pass, if the clocks satisfy the invariant. When a guard is satisfied, the corresponding transition can be taken. Transitions are instantaneous, and are additionally labelled with *clock resets* of the form $x := v$, which specify the value v that the clock x is reassigned to upon entering the target node.

The following is a simple timed automaton which has been extended with discrete probability distribution following [11][1].



The automaton models the process which repeatedly sends a packet every 4 to 5 ms, with 0.01 possibility of error, and then waits for 3 ms. It starts with the clock x set to zero in the node send, where it can remain until x reaches 5. At any time when x is between 4 and 5, it may either move to error with probability 0.01, or to wait with probability 0.99; in the latter case, x is reassigned to zero. It will remain in wait for exactly 3 ms before returning to send. A typical specification is: "if send then sometime receive within 8 ms with probability 0.9" It can be expressed in the logic PTCTL [11], a probabilistic variant of TCTL of [2], or SQTL [14]. Note that the timing is *exact*. By an automaton with *stochastic timing* we mean an automaton containing *clock resets* of the form $x := f(...)$ where f is a continuous time distribution[2], say exponential with rate 50 ms, resulting in a value drawn from the distribution f being assigned to the clock x.

Because of the inherent complexities of real-time systems, *validation* of quality of service is of paramount importance. A widely used method is *simulation*, see e.g. the generation of discrete event schedulers from logic [14] or Spades specifications [8]. The direction we take is that of *automatic verification through model checking* by developing the methods proposed in [2]. This involves building a finite quotient of the system, known as the region graph. Model checking

[1] By dropping the probabilities and duplicating guards (but not resets) of transitions joined by an arc, a standard timed automaton is obtained.

[2] We can retain the property that clocks increase uniformly with time by assigning to the clock $0 - c$ where c is the actual value drawn from distribution.

of purely timed systems can be thought of as (backward or forward) traversal of the underlying graph, modulo efficiency improvements such as *symbolic* methods which allow to manipulate sets of states, instead of individual states. The addition of discrete probability distributions (observe that non-determinism is present in timed automata in an essential way) corresponds to the construction of a Markov decision process, and solving an appropriate linear programming problem [5,4,9] to calculate the probabilities.

There is overwhelming evidence to support the case that the timed automata model and the associated tools can bring about concrete benefits. Several industrially-relevant case studies have been developed and model checked with Uppaal. In some, design errors have been detected which could not be found through testing [6]. More importantly though, some authors have specifically called for probabilistic modelling [15,7,8]. Thus, verification of quality of service through model checking would undoubtedly be useful.

3 Outline of Contribution

This paper summarises the work performed so far as part of the above project, which is available as [11,10,13,12] where we refer the interested reader for more details.

The starting point is the model of *discrete-probabilistic* timed automata introduced in [11], where also a model checking algorithm for the logic PTCTL is proposed. Unfortunately, this model checking algorithm is not readily implementable without further optimization due to high complexity (it is based on the region graph construction, which is exponential in the number of clocks).

Next two improvements based on *symbolic* model checking are presented, one generalising forward reachability [10] and the other backwards reachability [13]. The idea is to use symbolic representation (namely zones) to reduce the size of the quotient graph, and thus the size of the linear programming problem. It should be noted that, in contrast to the purely timed case, when verifying probabilistic timed automata we can no longer rely on the on-the-fly approach, since we cannot avoid the need to construct and solve the corresponding linear programming problem (which, in the worst case, is the size of the full region graph).

Finally, a *continuous-probabilistic* extension of the model which enables the modelling of stochastic timing is proposed in [12] together with a model checking algorithm.

References

1. L. Abeni and G. Butazzo. QoS guarantee using probabilistic deadlines. In *11th Euromicro Conference on Real-Time Systems (ECRTS99)*. 1999.
2. R. Alur, C. Courcoubetis, and D. Dill. Model-checking in dense real-time. *Information and Computation*, 104(1):2–34, 1993. Preliminary version appears in the Proc. of 5th LICS, 1990.

3. R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994. Preliminary version appears in Proc. 17th ICALP, 1990, LNCS 443.
4. C. Baier and M. Kwiatkowska. Model checking for a probabilistic branching time logic with fairness. *Distributed Computing*, 11:125–155, 1998.
5. A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Proceedings, FST&TCS*, volume 1026 of *Lecture Notes in Computer Science*, pages 499–513. Springer-Verlag, 1995.
6. H. Bowman, G. Faconti, J.-P. Katoen, D. Latella, and M. Massink. Automatic verification of a lip synchronisation protocol using UPPAAL. *Formal Aspects of Computing*, 10(5/6):550–575, 1998.
7. H. Bowman, G. Faconti, and M. Massink. Specification and verification of media constraints using UPPAAL. In *5th Eurographics Workshop on the Design, Specification and Verification of Interactive Systems, DSV-IS 98*.
8. P. D'Argenio, J.-P. Katoen, and E. Brinksma. Specification and analysis of soft real-time systems: Quantity and quality. In *Proceedings of the 20th IEEE Real-Time Systems Symposium,* Phoenix, Arizona, USA. IEEE Society Press, 1999. To appear.
9. L. de Alfaro. Computing minimum and maximum reachability times in probabilistic systems. In *Proc. 10th International Conference on Concurrency Theory (CONCUR'99)*, Lecture Notes in Computer Science. Springer-Verlag, 1999.
10. M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic verification of real-time systems with discrete probability distributions. Accepted for a Special Issue of *Theoretical Computer Science*.
11. M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic verification of real-time systems with discrete probability distributions. In J.-P. Katoen, editor, *Proceedings, ARTS'99*, volume 1601 of *Lecture Notes in Computer Science*. Springer-Verlag, 1999.
12. M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Verifying quantitative properties of continuous probabilistic timed automata. In *Proceedings, CONCUR'2000*, Lecture Notes in Computer Science. Springer-Verlag, 2000. To appear.
13. M. Kwiatkowska, G. Norman, and J. Sproston. Symbolic model checking of probabilistic timed automata using backwards reachability. Technical Report CSR-00-01, University of Birmingham, 2000.
14. A. Lakas, G. S. Blair, and A. Chetwynd. A formal approach to the design of QoS parameters in multimedia systems. In *Proceedings of the 4th International Workshop on Quality of Service*, 1996.
15. T. Stauner, O. Mller, and M. Fuchs. Using HyTech to verify an automotive control system. In O. Maler, editor, *HART'97, Proc. of the 1st International Workshop on Hybrid and Real-Time Systems*, volume 1201 of *Lecture Notes in Computer Science*. Springer-Verlag, 1997.