Quiescent Timed Automata: The Alternative for Quiescent Transition Systems

Marlène Hol
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
m.c.hol@student.utwente.nl

ABSTRACT

Quiescence is the absence of outputs in a labelled transition system. Quiescent Transition Systems (QTSs) are labelled transition systems that can model quiescence explicitly. Time outs in combination with QTSs are often used to determine quiescence in model-based testing. This is not yet done in a systematic and standardized way. This paper proposes a way to add quiescence to Timed Automata (TAs), the Quiescent Timed Automaton (QTA). TAs are transition systems extended with real-valued clocks. It is possible to determine quiescence in model-based testing via QTAs, because of the extra clock x that can be added to a TA to determine if quiescence is the case. The paper ensures that QTAs can be used for determining quiescence in testing, since it provides proof that it is an valid alternative for QTSs which does not alter the behavior of QTSs. Furthermore, a way is proposed to create a QTA from a labelled transition system and to directly create a QTA from a QTS and vice versa. QTAs as alternative for QTSs provide a foundation for a practical and systematic way of testing, which is now only done intuitively. The contribution of the QTA is therefore relevant for model-based testing with quiescence.

Keywords

Quiescence, Quiescent Transition System, Timed Automaton, Quiescent Timed Automaton

1. INTRODUCTION

Model-based testing is a common way of verifying the correctness of software. A lot of different forms or extensions of a Labelled Transition System (LTS) are used in model-based testing. An LTS is a model consisting of states and transitions which represent the different steps taken when executing a software program. In practical testing of LTSs it is currently hard to detect whether there really is an absence of outputs, or if the output is just delayed. Because of this issue, four different extensions for LTSs are presented in this paper: Input-Output Transitions Systems (IOTSs), Quiescent Transitions Systems (QTSs), Timed Automata (TAs) and Quiescent Timed Automata (QTAs). The first three extensions are already presented in earlier papers. The QTA is a new model first presented in this paper.

23rd Twente Student Conference on IT June 22, 2015, Enschede, The Netherlands.

First IOTSs are presented. An IOTS is an extension of a LTS, which distinguishes between actions that are initiated by the environment (input actions) and actions initiated by the system itself (output actions) [3]. This distinction is relevant for testing. Therefore in almost all relevant cases an IOTS is used for model-based testing instead of a regular LTS. In Figure 1 you can find an example of an IOTS, where a? is an input action and b! an output action.

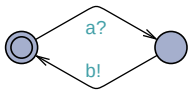


Figure 1: An example of an IOTS

QTSs are the next extension of LTSs introduced. When there is an absence of outputs in a transition system (for example an IOTS) this is called quiescence. Standard IOTSs do not contain an explicit way to represent quiescence. QTSs as introduced by Stokkink et al. [3] can represent quiescence explicitly with the quiescence label δ. In Figure 2 you can find an example of a QTS. Because the tool that created these models, Uppaal, does not contain the possibility to add the δ-symbol to a transition, this is represented by the text delta.

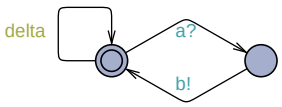


Figure 2: An example of a QTS

The third already known extension of a LTS is the TA. If quiescence is the case in a certain state when performing a test case and this is not foreseen in the specification, that particular test case should fail. In practice it is however hard to detect if an output is never given, i.e. if quiescence is the case. Time outs are often used in modern testing. Because time outs are used, an alternative for QTSs may be Timed Automata (TAs). TAs are finite automata, LTSs with a finite number of states, extended with a finite set of real valued clocks [1]. In Figure 3 you can find an example of a TA. Note that quiescence is not modeled in this example.

Practical testing is important to verify the correctness of software. As mentioned before, it is hard, if not impossible, in practical testing to determine whether quiescence,

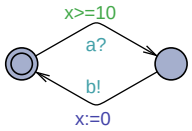


Figure 3: An example of a TA

the absence of outputs, is really the case. Therefore practical testing of LTSs, or QTSs when quiescence is modeled, is hard and there is not a standardized way to perform such tests.

Research shows that in current testing time outs are used to determine if quiescence is the case. Usage of time outs is however quite intuitive and they are not used in a systematic or standardized way, i.e. there is no real foundation for using time outs in combination with QTSs even though they are already used. TAs are, due their real valued clocks, capable of capturing timing in the transition systems. Using these clocks, it is therefore possible to tell after a certain amount of time if no output is given that quiescence is the case, i.e. time outs can be modeled and integrated in the transition systems when using TAs. Because of this property it becomes way more intuitive to use time outs for determining quiescence and TAs seem to model and test quiescence in a more systematical way. Therefore it is important to study TAs as an alternative for QTSs. If TAs can really be used as an alternative for QTSs, a foundation is made for a practical and systematic way of testing that is now only done intuitively.

Overview

To conduct tests with quiescence using TAs a few steps are needed beforehand. First, a way needs to be defined to integrate quiescence in TAs. No earlier research of TA presented this explicitly. If quiescence is not integrated in TAs there is no reason to explore the options of using them as an alternative for QTSs or to even make the comparison. In section 3 a solution of adding quiescence to TAs can be found, the Quiescent Timed Automaton (QTA). Secondly it is important that there is a way to create such a QTA from an IOTS. In this case, an IOTS is used instead of a LTS, because in most models of systems a distinction between input and output actions is necessary. This transformation is described in section 4. The third step of the research is the comparison of the QTS and the QTA created from the same IOTS. In [3] a way is already presented to transform a IOTS to a QTS. The comparison of the QTS and the QTA can ensure whether the QTA is indeed a reliable and formal alternative for the QTS. This is presented in section 5. In section 6 the transformation of the QTS to the QTA and vice versa can be found which is the last part of the research presented in this paper. These transformations are very useful for practical testing of your QTS without having the original IOTS or if you do not want to perform more than one transformation. The practical use of the QTA becomes way more realistic by the transformation presented in the section 6. All the transformations presented in this paper can be found in Figure 4.
Section 2 contains some background information about all the different transitions systems used in this paper. Some related work is also described in this section. This information is necessary to fully understand all the information presented in the paper. If already familiar with transition systems and timed automata this section can be skipped. All the next sections in this paper refer to a certain research question. The conclusion contains the answer to the main research question which can be answered by combining the information of all previous sections.

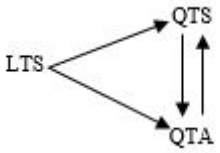


Figure 4: The transformations presented in this paper

Research questions

All research questions presented here are answered in all upcoming sections. The main research question is the first question presented and focuses on the overall idea of finding a way to use a TA as an alternative for a QTS. The other research questions focus on the steps necessary to ensure that TAs are an alternative for QTSs.

1. How can timed automata where quiescence is added to be used in a way that their behavior and structural properties are preserved and timed automata can be used as an alternative for quiescent transition systems?

2. How can quiescence be added to a timed automaton in such a way that the timed automaton keeps all its original properties and behavior?

3. How can a timed automaton be created from a labelled transition system in a way that it has the same behavior and what are the limitations of this transformation?

4. To what extent are a timed automaton and a quiescent transition system created from the same labelled transition system equivalent when it comes to behavior and test results when performing a certain test?

5. How can a timed automaton be created from a quiescent transitions system or a quiescent transitions system from a timed automaton in a way that it has the same specifications and what are the limitations of this transition?

2. BACKGROUND

2.1 Labelled Transition Systems

Definition 1. A labelled transition system (LTS) is a quadruple A = <S, S^0, L $\cup$ {τ}, $\rightarrow$> such that:

- S is a set of states
- $S^0 \subseteq$ S is a non-empty set of initial states
- L is a set of labels where each element represents a different action, and where τ is an internal unobservable action
- $\rightarrow \subseteq S \times L^\tau \times S$ is the transition relation, where $L^\tau = L \cup \{\tau\}$

With a standard LTS there is no distinguishment between actions that are initiated by the environment, input actions, and actions that are initiated by the system itself, output actions. Therefore an extension of the standard LTS is created, the input-output transition system (IOTS).

Definition 2. An IOTS is a quintuple A =<S, S^0, L^I , $L^O \cup \{\tau\}$, →> such that:

- S is a set of states
- $S^0 \subseteq S$ is a non-empty set of initial states
- L^I is a set of input labels where each element represents a different input action
- L^O is a set of output labels where each element represents a different output action, and where τ is an internal unobservable action
- $\rightarrow \subseteq S \times L^\tau \times S$ is the transition relation, where $L^\tau = L^I \cup L^O \cup \{\tau\}$

IOTSs are the basis for all the transitions systems presented in this paper. Three important standard operations defined for IOTSs are determinisation, parallel composition and action hiding [3].

2.2 Quiescence and Quiescent Transition Systems

Quiescence is the absence of outputs. Quiescence was introduced for the first time by Vaandrager [7]. Tretmans [5] then introduced the concept of repetitive quiescence and made it possible to continue testing (even in quiescent states) by introducing the suspension automata. This was partly a solution to the quiescence issues, but there were still some shortcomings. Quiescence is for example not treated as a first-class citizen and basic operators like parallel composition and action hiding are not defined for suspension automata. Therefore a new theory for quiescence is presented by Stokkink et al. [3]: the quiescent transition systems (QTSs).

First a formal definition of quiescent states is given. This definition is relevant in order to understand the formal definition of the QTS, which is also defined here.

Definition 3. A state s of a transition system A = <S_a, S^0_a, L^I_a , L^O_a, $\rightarrow_a$> is called quiescent when $\forall$ (s,a,s′) $\in$ $\rightarrow_a$ then a $\notin L^O \cup \{\tau\}$. The set of quiescent states of the transition system A will be referred to as q(A).

Definition 4. A QTS is an IOTS A =<S, S^0, L^I , $L^O \cup \{\tau\} \cup \{\delta\}$, →> such that

- S is a set of states
- $S^0 \subseteq S$ is a non-empty set of initial states
- L^I is a set of input labels where each element represents a different input action
- L^O is a set of output labels where each element represents a different output action, where τ is an internal unobservable action, and where δ is a special output label that is used to denote the observation of quiescence
- $\rightarrow \subseteq S \times L^\delta_\tau \times S$ is the transition relation, where $L^\delta_\tau = L^I \cup L^O \cup \{\delta\} \cup \{\tau\}$

The standard operations determinisation, parallel composition and action hiding are defined for QTSs [3]. QTSs can be created from an IOTS by the process called deltafication. The following well-formed rules are defined for QTSs:

1. Quiescence should be observable
2. No outputs after quiescence
3. Quiescence does not enable new behavior
4. Continued quiescence preserves behavior

QTSs can be convergent and divergent. They are called divergent if there is an infinite path that only contains τ transactions. Stokkink et al. [3] first presented the results only for input-enabled and convergent QTSs. In [4], the author however presents Divergent Quiescent Transition Systems, DQTSs, which can also handle divergent IOTSs. The research in this paper will only focus on convergent and input-enabled IOTSs and QTSs. Research conducted with DQTSs is future work and is not included in this paper.

2.3 Timed Automata

This subsection first presents the formal definition of a timed automaton given by Alur et al. [1]. The symbols used in this definition are adapted to the symbols used by Stokkink et al. [3] and are therefore different from the original symbols used by Alur et al.

Definition 5. A timed automaton is a quintuple <S, S^0, L ∪ {τ}, C, →> such that

- S is a set of states
- S^0 is a set of non-empty initial states
- L is a set of labels where each element represents a different output action, and where τ is an internal unobservable action
- C is a finite set of real-valued clocks
- $\rightarrow$ is a transition which is defined as (s, a, s′, λ, μ) $\in S \times L^\tau \times S \times \Phi(C) \times 2^C$, where $L^\tau = L \cup \{\tau\}$, λ is the set of clocks that need to be reset with this transition, and μ is a clock constraint over the clock C.

Earlier research is conducted on timed automata. The two papers on this topic that are most important for this research are shortly described here. First, the paper by Alur et al. [1] which focuses on the fundamental theory of TAs. This paper introduces TAs and gives the first formal definition of TAs. This is followed up by a way to check emptiness and intractable problems. Finally, deterministic timed automata are defined and there is a topic on the verification of timed automata. The second paper by Bengtsson et al. [2] is about using TAs for modeling and verification of real time systems. In this paper TAs are presented which is followed up by the symbolic semantics and the verification of TAs. Next, the algorithm and data structures of TAs are presented and in the end the use of UPPAAL for TAs. UPPAAL is a tool that is used to model and verify TAs [6]. This tool is also used in this research to create the images of transitions systems and (Q)TAs used in this paper.

3. QUIESCENT TIMED AUTOMATA

In the previous section TAs are already introduced, but quiescence for TAs is not specified in earlier research. To model quiescence using TAs, Quiescent Timed Automata (QTAs) are introduced.

Definition 6. A Quiescent Timed Automata is a TA A =<S, S^0, L^I, $L^O \cup \{\tau\} \cup \{\delta\}$, C, $\rightarrow$>, such that:

- S is a set of states. $\forall$ s $\in$ S there is an invariant x $\leq$ t.
- $S^0 \subseteq$ S is a non-empty set of initial states.
- L^I is a set of input labels where each element represents a different input action, which are recognized by the ? at the end.
- L^O is a set of output labels where each element represents a different output action that are recognized by the ! at the end, where τ is an internal unobservable action, and where δ is the label that represents quiescence explicitly.
- C is a finite set of clocks which contains the extra clock x.
- $\rightarrow \subseteq S \times L \times S \times \Phi(C) \times 2^C$ is the transition relation, where $\mu \in \Phi(C)$ is the set of clock constraints over C and $\lambda \in 2^C$ is the set of clocks that need to be reset together with this transition. $\forall$ (s,a,s$'$,μ,λ) $\in \rightarrow$ then x $\in \lambda$. $\forall$ s $\in$ q(A) then $\exists$ (s,a,s$'$,μ,λ) $\in \rightarrow$ where a $\in \{\delta\}$, and x == t $\in \mu$.

Every QTA has a special clock x which measures whether a state s is quiescent. This clock x is only used to determine whether quiescence is the case and can not be used for any other purpose. Every QTA also has a fixed time t after which a state is said to be quiescent. A distinction is made between quiescent and non-quiescent states. In non-quiescent states the invariant makes sure that if after the time t no transition happened, the state is marked as quiescent. Since this is a non-quiescent state this is unwanted behavior and the invariant makes it possible to detect erroneous behavior in (the model of) a system. In quiescent states there is, besides the invariant x $\leq$ t, an outgoing transition with the δ label and the constraint that x == t. Because of the combination of the invariant and the clock constraint that the δ-transition needs to happen at a time t, it is ensured that quiescence is always noticed in quiescent states because of the δ-transition which always happens at the time t. Note that to get a correct working of the QTA every transition needs to reset the clock x.

The time t after which a state is considered to be quiescent, can be determined by the user. This can depend on the time used by the system, the preferences of the user, or the kind of system that needs to be tested. When the time t is chosen incorrectly this has some risks for the system. First, the time t chosen can be too long. In that case, in quiescent states the special transitions with the quiescence label δ can only happen after that long time t and in that time t another input transition might already happened and quiescence is never noticed. In QTSs, that use the same quiescence label, it is assumed that the δ-transition happens. In QTAs this depends on the clock. If the time t is too long, quiescence can stay unnoticed and a QTA will therefore have a different behavior than the QTS and can not be used as an alternative for the QTS.

The second risk is that the time t chosen is too short. In non-quiescent states the following invariant is added: you can only be in a certain state if the clock x $\leq$ t. If the clock x is greater than t, the state is marked as quiescent and the system is said to not work properly. If the time t chosen is too short, non-quiescent states can be marked as quiescent too soon and the system can be labelled as erroneous. This is unwanted behavior. Another side-effect of the time t chosen too short is that in a quiescent state the δ-transition happens real soon and even multiple times before another transition might happen. This is not per definition erroneous behavior, but it is unwanted behavior. When determining the time t, the user should keep all of these risks in mind. This gives great responsibility to the user. It is impossible to set a standard for this, since this time heavily depends on the behavior of the system. It is assumed however that the time t is chosen in such a way that it should never happen that quiescence is noticed too soon or not at all.

4. FROM IOTS TO QTA

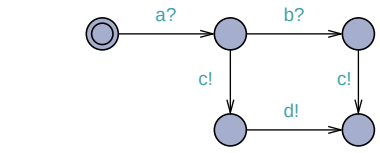
IOTSs are most commonly used when modeling a system. In order to perform the necessary testing on such models, the notion of quiescence needs to be made explicit. One way to do this is proposed in the previous section: the QTA. In this section we present a way to transform an IOTS to a QTA by adding a new clock x to the IOTS, and formulate constraints on this clock x for the transitions and states of the original IOTS, to make the notion of quiescence explicit. An IOTS can be tested in a practical way after this transformation. Notice that this clock x is only used to determine quiescence and is not used for any other purposes.

Definition 7. Given an IOTS A = <S_a, S_a^0, L_a^I ,$L_a^O \cup \{\tau\}$, $\rightarrow_a$> we define the transition of A to the QTA B C(A) = <S_b, S_b^O, L_b^I, $L_b^O \cup \{\tau\} \cup \{\delta\}$, C_b, $\rightarrow_b$> where:

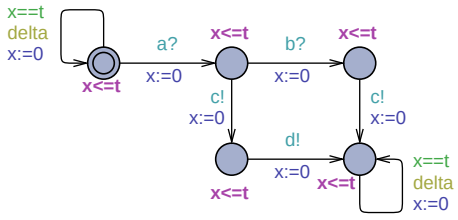
- C_b consists of the extra clock x
- $S_a = S_b$ and $\forall$ s $\in S_b$ an invariant is added to s that x $\leq$ t
- $S_a^0 = S_b^0$
- $L_a^I = L_b^I$
- $L_a^O = L_b^O$
- $\rightarrow_b = \{(s, a, s', \lambda, \mu) \in S_b \times L_b^\tau \times S_b \times \Phi(C) \times 2^C \mid (s,a,s') \in \rightarrow_a \wedge \mu = \emptyset \wedge \lambda = \{x\})\} \cup \{(s, \delta, s, \lambda, \mu) \in S_b \times \{\delta\} \times S_b \times \Phi(C) \times 2^C \mid s \in q(B) \wedge \mu = \{x == t\} \wedge \lambda = \{x\})\}$, where $L_b^\tau = L_b^I \cup L_b^O \cup \{\tau\}$

An example of the transformation of an IOTS A to the corresponding QTA can be found in Figure 5. No value is given to the time t after which quiescence in a state is determined.

A difference between two kind of transitions from state s to state s$'$ is made with the transformation of an IOTS to the QTA: the transition was already part of the IOTS A, or state s is quiescent and the special quiescence transition occurs. For the first possibility the transition is an input, output or internal action and the addition is made that clock x needs to be reset. There is no clock constraint for such a transition. In the second possibility the state s is quiescent and therefore the clock should determine this after the time t. For this purpose an extra transition with the special quiescence label δ is added. This transition has the clock constraint that the clock x has to be equal to t and after this transition the clock x resets. Because of the



(a) The IOTS A



(b) The QTA B

Figure 5: The transformation of the IOTS A to the QTA B

combination of the invariant in the quiescent states that x needs to be smaller or equal to t and the constraint for the δ-transition that x needs to be equal to t it is ensured that the delta transition always occurs in a quiescent state.

The definition of the transition relation covers the notice of quiescence in quiescent states. QTAs have the great benefit to also notice quiescence in non-quiescent states and therefore detect incorrect behavior in (the model of) a system. To ensure this happens the invariant is also added to every non-quiescent state that the clock x needs to be smaller or equal to t. After adding the delta transition to quiescent states and adding this invariant to both the quiescent and the non-quiescent states a correct QTA is created from an IOTS where quiescence can be noticed in every state. Note that this only works correctly if with every transition the clock x resets.

This transformation to a QTA ensures that the behavior of the original IOTS is not altered. In each state all the input, output, and internal actions of the IOTS are also possible in the QTA, without any restrictions on time. Adding the invariant to non-quiescent states can however create some differences in the behavior of an IOTS and a QTA created from this IOTS, since there is no notion of time in the IOTS. In the IOTS it is assumed an input, output, or internal transition happens. In a QTA a transition can only happen if the invariant allows it. However, it is assumed that the time t after which such a non-quiescent state is marked quiescent is chosen in such a way that all transitions are still possible under normal circumstances. With this assumption the invariant does not influence the behavior of the created QTA in such a way that it is different from the behavior of the IOTS.

The aim of this research is to use QTAs as an alternative for QTSs when testing with quiescence. To ensure this, in the next section comparisons are made between the QTS and the QTA created from the same IOTS. To be sure no transformation rules are needed for this, the QTA uses the same special quiescence label δ as proposed by Stokkink et al. [3]. The quiescence label is necessary to make explicit that quiescence was really the case.

5. COMPARISON OF QTS AND QTA

As mentioned before, the aim of this research is to use QTAs as an alternative for QTSs. In order to ensure QTAs are a valid alternative for QTSs, the specifications and the behavior of both transition systems have to be equal. In this section several comparisons of a QTA and a QTS created from the same IOTS are shown. An example can be found in Figure 6. No value is given to the time t after which quiescence in a state is determined. This example will be referred to in the upcoming subsections.

5.1 Trace equivalence

The first aspect that is relevant to establish that QTAs are an alternative for QTSs is to determine if both systems are trace equivalent. Before ensuring that both systems are trace equivalent the definitions of trace and trace equivalence and the definition of path is given.

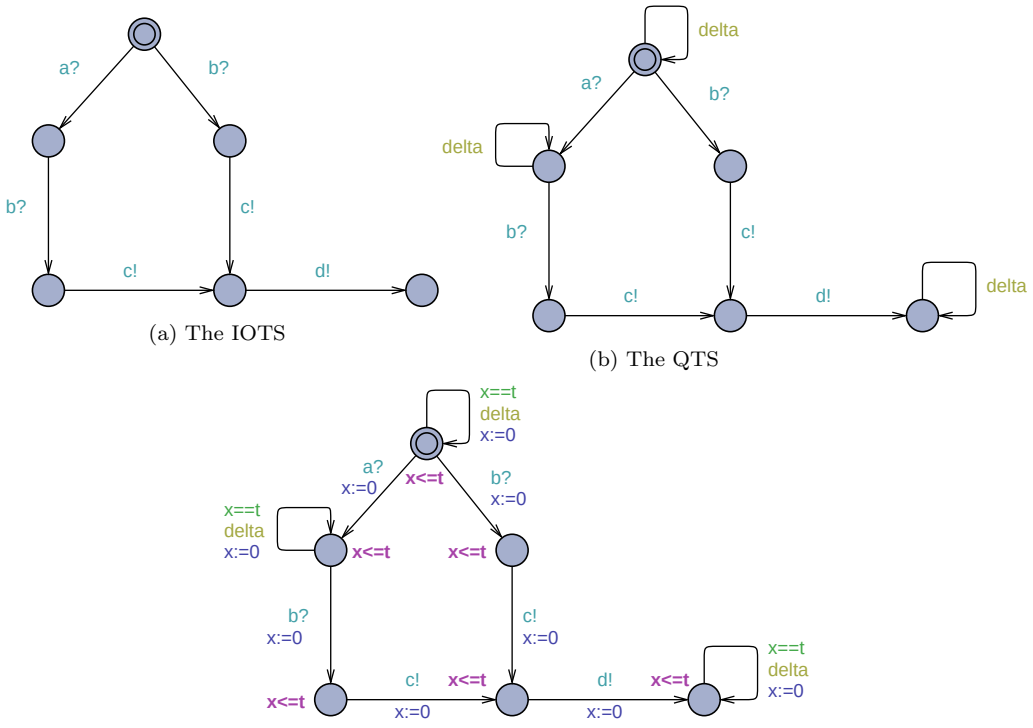
Definition 8. A trace $\sigma = a_1 a_2 \dots a_n$ is the sequence of actions executed by a transition system A, where the τ-actions are removed. The length of the trace is denoted by $|\pi|$ and is the number of actions occurring in the trace. The set of all traces of A is denoted by Traces(A). Two transition systems A and B are called trace equivalent when traces(A) = traces(B).

Definition 9. A path is a sequence $\pi = s_0 a_1 s_1 a_2 \dots a_n s_n$ of a transition system A such that $\forall\ 1 \leq i \leq n$ then $(s_{n-1}, a_n, s_n) \in \rightarrow_A$. For this path trace($\pi$) = $a_1 a_2 \dots a_n$. The set of all traces of A is denoted by paths(A).

Theorem 1. Let A be an IOTS = $<S_a, S_a^0, L_a^I, L_a^O \cup \{\tau\}, \rightarrow_a>$, B be the QTS = $<S_b, S_b^0, L_b^I, L_b^O \cup \{\tau\} \cup \{\delta\}, \rightarrow_b>$ created from the IOTS A by deltafication as described in [3], and C the QTA = $<S_c, S_c^0, L_c^I, L_c^O \cup \{\tau\} \cup \{\delta\}, C_c, \rightarrow_c>$ created from the IOTS A by the algorithm described in Definition 7. Then traces(B) = traces(C).

Proof. First, we show that traces(B) $\subseteq$ traces(C). Let $\sigma = a_1 a_2 \dots a_n \in$ traces(B). We need to show that $\sigma \in$ traces(C). By Definition 9, we know that there is a path $\pi = s_0 a_1 s_1 a_2 \dots a_n s_n$. QTS B and QTA C are only trace equivalent if the trace of π is in both B and C. We define $L_a = L_a^I \cup L_a^O \cup \{\tau\}$, $L_b = L_b^I \cup L_b^O \cup \{\tau\} \cup \{\delta\}$, and $L_c = L_c^I \cup L_c^O \cup \{\tau\} \cup \{\delta\}$. By definition of deltafication [3] we know that $L_a \cup \{\delta\} = L_b$, $S_a = S_b$, and $S_a^0 = S_b^0$. By Definition 7 we know that $L_a \cup \{\delta\} = L_c$, $S_a = S_c$, and $S_a^0 = S_c^0$. Because of these definitions we also know that $\rightarrow_a \subseteq \rightarrow_b$ and $\rightarrow_a \subseteq \rightarrow_c$. Therefore all the input, output and internal transitions in the IOTS, the QTS and the QTA are the same (1). From (1) it can also be concluded $\forall\ a_n \in \sigma$ we know that if $a_n \notin L_a$ then $a_n \in \{\delta\}$. For both the QTS and the QTA Definition 3 is used for quiescent states. It is known $S_a = S_b$ and $S_a = S_c$ and therefore $S_b = S_c$. Because of this, and the definition of quiescent states we know q(B) = q(C). Besides this we know that $\forall\ s \in$ q(B) the delta transition happens and thus $\forall\ a_n \in \{\delta\}$ that $s_n \in$ q(B). Because q(B) = q(C) also $s_n \in$ q(C) and therefore that the transition with label $a_n \in \rightarrow_c$ (2). By Definition 6 we know that for all transitions in the QTA the clock constraint and the clocks that need to be reset with those transitions need to be defined. By Definition 7 we know that when creating the QTA from the IOTS the clock x is added and that the clock constraint and the clocks that need to be reset are added to the transitions, but that this does not influence the transitions possible and even make sure that the delta transition indeed happens as is assumed with QTSs. Therefore it can be concluded from 1 and 2 that $\forall\ (s_{n-1}, a_n, s_n) \in \pi$ that there $\exists\ (s_{n-1}, a_n, s_n, \mu, \lambda) \in \rightarrow_c$ and thus that $\sigma \in$ traces(C).

Now we have to show that traces(C) $\subseteq$ traces(B). This proof is symmetric to the first part of the proof and therefore we know the traces(C) $\subseteq$ traces(B).



(a) The IOTS

(b) The QTS

(c) The QTA

Figure 6: The transformation of the IOTS A to the according QTS and QTA

Since it is shown that traces(B) $\subseteq$ traces(C) and traces(C) $\subseteq$ traces(B) it can now be concluded that traces(B) = traces(C) and thus that the QTS B and the QTA C created from the same IOTS A are trace equivalent. $\square$

There are two important issues to notice when exploring trace equivalence for the QTA and the QTS due to the addition of the real-valued clock in the QTA. First there can be a difference in the traces of the QTA and the QTS because of the invariant added to all non-quiescent states of the QTA. With the QTS one can assume that an input, output, or internal action happens. QTAs have time restrictions, which brings the possibility that an input, output or internal action does not happen before the time of clock x is already bigger than the invariant allows, and the state is marked as quiescent before the input, output, or internal action can happen. In this case the QTA and QTS are not trace equivalent. However it can be assumed that the user chooses a time long enough that every action can occur. With this assumption the QTA and the QTS are trace equivalent.

The second issue that should be taken into consideration is that QTSs do not have any (timing) constraints on when a δ-transition should happen. It can therefore be that in a QTS two δ-transitions appear shortly after another or that the δ-transition happens after a long period of time. In the QTA there is the timing constraint, which makes sure that a delta-transition happens after a certain time. The possibility that two delta-transitions happen shortly after another is therefore not there (assuming that the time t is not chosen too short). When performing brute-force testing it can therefore happen that there are an unequal number of δ-transitions in a certain state s. This is only the case when performing practical testing for trace equivalence on both the transition systems. Both the QTS and the QTA are however theoretical models and therefore this is not considered as an issue. When practical testing with the QTA is performed, it is for the user of a QTA to notice this aspect when determining the time t after which a state is marked as quiescent.

5.2 Well-formed rules

In section 3.1 of the paper by Stokkink et al. [3] the so called well-formed rules are presented. If a QTS obeys to all these rules, a QTS is called well-formed. In order to determine if the QTA is an alternative for QTSs, it is necessary that the QTA also holds to these rules when a QTS is well-formed. In order to make this comparison more readable, the well-formed rules are included in this section. These well-formed rules are stated in the notation of the paper by Stokkink et al. [3]

- **Rule R1** (Quiescence should be observable): if s is quiescent, then $s \xrightarrow{\delta}$.
- **Rule R2** (No outputs after quiescence): if $s \xrightarrow{\delta} s'$, then s′ is quiescent.
- **Rule R3** (Quiescence does not enable new behavior): if $s \xrightarrow{\delta} s'$, then traces(s′) $\subseteq$ traces(s).
- **Rule R4** (Continued quiescence preserves behavior): if $s \xrightarrow{\delta} s'$ and $s' \xrightarrow{\delta} s''$, then traces(s′) = traces(s″).

Theorem 2. Let A be an IOTS = <S_a, S_a^0, L_a^I ,$L_a^O \cup \{\tau\}$, $\rightarrow_a$>, B be the QTS = <S_b, S_b^0, L_b^I , $L_b^O \cup \{\tau\} \cup \{\delta\}$, $\rightarrow_b$> created from the IOTS A by deltafication as described in [3] , and C the QTA = <S_c, S_c^0, L_c^I, $L_c^O \cup \{\tau\} \cup \{\delta\}$, C_c, $\rightarrow_c$> created from the IOTS A by the algorithm described in Definition 7. Then if B is well-formed, then C is well-formed.

Proof. Let B be a QTS that is well-formed. We need to show that C is also well-formed. We define $L_b = L_b^I$

$\cup$ L_b^O $\cup$ $\{\tau\}$ $\cup$ $\{\delta\}$, and L_c = L_c^I $\cup$ L_c^O $\cup$ $\{\tau\}$ $\cup$ $\{\delta\}$. From Theorem 1 we know that S_b = S_c, S_b^0 = S_c^0, L_b = L_c and we know that B and C are trace equivalent. For the determinisation of quiescent states of both the QTS and the QTA Definition 3 is used. Because of this and because S_b = S_c we know that q(B) = q(C). Because B and C are trace equivalent and because q(B) = q(C) we know that if Rule R1, R2, R3 and R4 apply to the QTS B it also applies to the QTA C. Because q(B) = q(C) all the quiescent states of B and C are the same and because they are trace equivalent we know that all the δ transitions that happen in B also happen in C. If B applies to the well-formed rules it is therefore only possible that the QTA C also applies to these rules. □

Note that the clock x added to the QTA to determine quiescence does not influence the behavior of the QTA in a way that the well-formed rules do not apply anymore. As mentioned before, it is possible that the time t after which a state is said to be quiescent is chosen in such a way that the state is too said to be quiescent and certain transitions can not happen. It is however assumed that the time t is chosen in such a way that this is not the case and that the clock does not influence the behavior of the QTA. Therefore the well-formed rules do apply on the QTA when they apply on the QTS.

This section showed that a QTS and a QTA created from the same IOTS are trace equivalent and that if the well-formed rules as defined by [3] apply on the QTS, they also apply on the QTA. Therefore, QTAs are a valid alternative for QTSs and the behavior of QTSs is not altered when using QTAs instead.

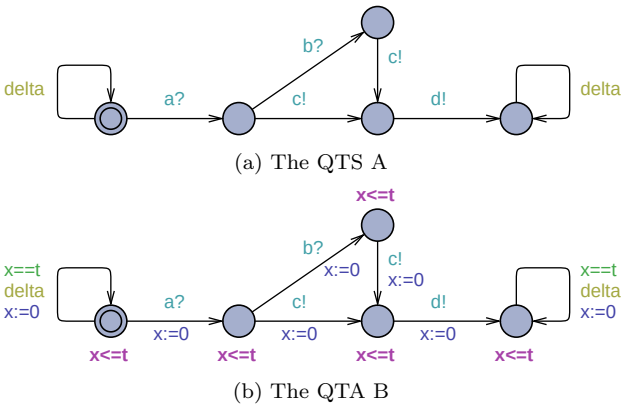
6. TRANSFORMATIONS

In the previous section it is already proven that a QTS and a QTA created from the same IOTS are trace equivalent and that if the well-formed rules apply on the QTS, they also apply on the QTA. This makes clear that it is also possible to create a QTA from a QTS (and vice versa) without using the original IOTS. In this section we first present the transformation from a QTS to a QTA. This transformation basically consists of adding the clock x and the constraints belonging to this clock to the QTS. This section also contains the proof that a QTS and the QTA created from this QTS are trace equivalent and that if the well-formed rules apply on the QTS, they also apply on the QTA. In the next subsection we present the transformation from the QTA to the QTS. With this transformation the clock x and all its belonging constraints are removed.

6.1 From QTS to QTA

Definition 10. Given the QTS A = <S_a, S_a^0, L_a^I ,L_a^O $\cup$ $\{\tau\}$ $\cup$ $\{\delta\}$, $\rightarrow_a$> we define the transition of A to the QTA B T(A) = <S_b, S_b^O, L_b^I, L_b^O $\cup$ $\{\tau\}$ $\cup$ $\{\delta\}$, C_b, $\rightarrow_b$> where:

- C_b consists of the extra clock x
- S_a = S_b and $\forall$ s $\in$ S_b the invariant is added to s that x $\leq$ t
- S_a^0 = S_b^0
- L_a^I = L_b^I
- L_a^O = L_b^O
- $\rightarrow_b$ = $\{$(s, a, s′, λ, μ) $\in$ S_b $\times$ L_b^τ $\times$ S_b $\times$ Φ(C) $\times$ 2^C | (s,a,s′) $\in$ $\rightarrow_a$ $\wedge$ μ = $\{\}$ $\wedge$ λ = $\{x\}$)$\}$ $\cup$ $\{$(s, δ, s′, λ, μ) $\in$ S_b $\times$ $\{\delta\}$ $\times$ S_b $\times$ Φ(C) $\times$ 2^C | (s,δ,s′) $\in$ $\rightarrow_a$ $\wedge$ μ = $\{x == t\}$ $\wedge$ λ = $\{x\}$)$\}$, where L_b^τ = L_b^I $\cup$ L_b^O $\cup$ $\{\tau\}$



(a) The QTS A

(b) The QTA B

Figure 7: The tranformation of the QTS A to the QTA B

Figure 7 shows an example of the QTS A and the corresponding QTA B which is created from the QTS A by the algorithm described in Definition 10. No value is given to the time t after which quiescence in a state is determined.

Contrary to IOTSs QTSs already contain the transition with the special δ label for quiescence. Therefore for $\rightarrow_b$ a distinction is now made between a transition that is already labelled with the δ label or a transition that is labelled with any other label. With every transition that occurs in a QTA the clock x needs to be reset because that is the only way possible to detect if quiescence is the case in that a state. Transitions with the special quiescence label δ also have the clock constraint that when the clock x is exactly t seconds the δ transition should happen. In combination with the invariant added to all states it is ensured that the delta transitions happen in all quiescent states and that it is determined that this state is quiescent. Because of this constraint a distinction is made between the δ transition and all other transitions when defining $\rightarrow_b$.

In order to determine if the QTA created from a QTS as described by Definition 10 is an alternative for the QTS it needs to be ensured that also in this case the QTS and QTA are trace equivalent and that the well-formed rules also apply on the QTA when they apply on the QTS.

Theorem 3. Let A be a QTS = <S_a, S_a^0, L_a^I ,L_a^O $\cup$ $\{\tau\}$ $\cup$ $\{\delta\}$, $\rightarrow_a$> and B the QTA = <S_b, S_b^O, L_b^I, L_b^O $\cup$ $\{\tau\}$ $\cup$ $\{\delta\}$, C_b, $\rightarrow_b$> that is created from A as described in Definition 10. Then traces(A) = traces(B).

PROOF. First we show that traces(A) $\subseteq$ traces(B). Let σ = $a_1 a_2 \ldots a_n$ $\in$ traces(A). We need to show that σ $\in$ traces(B). By Definition 9 we know that there is a path π = $s_0 a_1 s_1 a_2 \ldots a_n s_n$. If A and B are trace equivalent then π $\in$ paths(B). We define L_a = L_a^I $\cup$ L_a^O $\cup$ $\{\tau\}$ $\cup$ $\{\delta\}$ and L_b = L_b^I $\cup$ L_b^O $\cup$ $\{\tau\}$ $\cup$ $\{\delta\}$. By Definition 10 we know that S_a = S_b, S_a^0 = S_b^0, L_a = L_b. By Definition 10 we can also conclude that all the transitions possible in the QTS can also happen in the QTA. For all these transitions in the QTA the clock constraint on the extra clock x and if clock x needs to be reset is added. By Definition 10 we know that the extra added clock x does not influence the transition possible and that no extra transitions are added to the QTA when creating it from the QTS. Because of this we know that $\forall$ (s_{n-1},a_n,s_n) $\in$ π that there $\exists$ (s_{n-1},a_n,s_n,μ,λ)

$\in \rightarrow_b$ and thus that $\sigma \in$ traces(B). Therefore traces(A) $\subseteq$ traces(B).

Now we have to show that traces(B) $\subseteq$ traces(A). Let $\sigma = b_1 b_2 \ldots b_n \in$ traces(B) and let the path $\pi = s_0 b_1 s_1 b_2 \ldots b_n s_n$ be the path from σ. We need to show that $\sigma \in$ traces(A). By Definition 10 we know that all the transition of the QTS A are also in the QTA B and that no new transitions are added when creating the QTA. To the transitions of the QTA the clock constraint on the added clock x and if clock x needs to be reset is added, but by Definition 10 we know that this does not alter the transitions possible. Because of this we know that $\forall$ (s_{n-1},b_n,s_n) $\in \pi$ that there $\exists$ (s_{n-1},b_n,s_n) $\in \rightarrow_a$ and therefore $\sigma \in$ traces(A). Therefore traces(B) $\subseteq$ traces(A).

Since traces(A) $\subseteq$ traces(B) and traces(B) $\subseteq$ traces(A) it can be concluded that traces(A) = traces(B). The QTS A and the QTA B created from QTS A by Definition 10 are therefore trace equivalent. □

Theorem 4. Let A be a QTS = <S_a, S_a^0, L_a^I ,$L_a^O \cup \{\tau\} \cup \{\delta\}$, $\rightarrow_a$> and B the QTA = <S_b, S_b^O, L_b^I, $L_b^O \cup \{\tau\} \cup \{\delta\}$, C_b, $\rightarrow_b$> that is created from A as described in Definition 10. If the well-formed rules as described in section 5.2 apply to the QTS A, then they also apply to the QTA B.

Proof. Let A be a QTS that is well-formed. We need to show that the QTA B is also well-formed. We define $L_a = L_a^I \cup L_a^O \cup \{\tau\} \cup \{\delta\}$ and $L_b = L_b^I \cup L_b^O \cup \{\tau\} \cup \{\delta\}$. By Definition 10 we know that $S_a = S_b$, $S_a^0 = S_b^0$, $L_a = L_b$ and that A and B are trace equivalent. Because $S_a = S_b$ and because for both QTS A and QTA B Definition 3 is used to define quiescent states we know that q(A) = q(B). Therefore and because of the trace equivalence we know that all the well-formed rules apply also on the QTA B if they apply on the QTS A. □

The QTA created from the QTS is a valid alternative for the QTS since both transition systems are trace equivalent and when the QTS applies to the well-formed rules then the QTA also applies to these rules. The main difference between these two transition systems is that with the QTS you can assume transitions happen but with the QTA the clock determines if an action can happen (or can not happen). This can create some differences between both transition systems when performing tests. In general it however assumes that the time t is chosen in such a way, this is not the case. Therefore the QTA created from a QTS is a good alternative for this QTS.

6.2 From QTA to QTS

Earlier in this section it is explained that QTAs can be created from QTSs by adding the extra clock x. In this subsection the exact opposite is presented by removing the clock x from a QTA to create a QTS.

Definition 11. Given the QTA B = <S_b, S_b^O, L_b^I, $L_b^O \cup \{\tau\} \cup \{\delta\}$, C_b, $\rightarrow_b$> we define the transition of B to the QTS C T(B) = <S_c, S_c^0, L_c^I ,$L_c^O \cup \{\tau\} \cup \{\delta\}$, $\rightarrow_c$> where:

- the extra clock x is removed
- $S_b = S_c$ and $\forall$ s $\in S_c$ the invariant x $\leq$ t is removed
- $S_b^0 = S_c^0$
- $L_b^I = L_c^I$
- $L_b^O = L_c^O$
- $\rightarrow_c = \{(s, a, s') \in S_c \times L_\tau^\delta \times S_c \mid (s,a,s') \in \rightarrow_b \wedge$ the clock constraint is removed $\wedge$ the set of clocks that need to be reset is removed$\}$, where $L_\tau^\delta = L_c^I \cup L_c^O \cup \{\tau\} \cup \{\delta\}$

In order to ensure that the QTS created from a QTA is a valid alternative for the QTA it once again needs to be ensured that both transition systems are trace equivalent and that if the well-formed rules apply on the QTA they also apply on the QTS.

Theorem 5. Let A be a QTA = <S_a, S_a^O, L_a^I, $L_a^O \cup \{\tau\} \cup \{\delta\}$, C_a, $\rightarrow_a$> and B the QTS = <S_b, S_b^0, L_b^I ,$L_b^O \cup \{\tau\} \cup \{\delta\}$, $\rightarrow_b$> that is created from A as described in Definition11. Then traces(A) = traces(B).

Proof. This proof is similar to the proof of Theorem 3. From this we can conclude that Traces(A) = Traces(B). □

Theorem 6. Let A be a QTA = <S_a, S_a^O, L_a^I, $L_a^O \cup \{\tau\} \cup \{\delta\}$, C_a, $\rightarrow_a$> and B the QTS = <S_b, S_b^0, L_b^I ,$L_b^O \cup \{\tau\} \cup \{\delta\}$, $\rightarrow_b$> that is created from A as described in Definition11. If the well-formed rules as described in section 5.2 apply to the QTA A, then they also apply to the QTS B.

Proof. This proof is similar to the proof of Theorem 4. From this we can conclude that if the well-formed rules apply on the QTA A, they also apply on the QTS B. □

The proofs of Theorem 3 and Theorem 4 are based on the fact that all the transitions are in both transition systems and that for both transition systems Definition 3 is used to determine quiescent states. This is also the case in the transformation from QTA to QTS. Therefore it is certain that the proofs are similar and that the QTS created from the QTA and the QTA are trace equivalent and that if the well-formed rules apply on the QTA they also apply on the QTS.

For this transformation it also counts that the difference between these two systems is that in the QTA the clock determines whether an action can happen or can not happen. In transition systems without integrated clocks like the created QTS, it is assumed that all transitions (can) happen at a certain point. This can create differences between two models, but as explained before, it is assumed that the time t after which the clock x notices quiescence is chosen in such a way that this in not the case. Therefore the QTS created from a QTA is a valid alternative for the QTA.

Note that the QTS C presented in Definition 11 is exactly the same as the QTS A presented in Definition 10. These two definitions show that the transformation from the QTS to the QTA is reversible and that it does not influence the behavior of the system. This is very important when you look at the purpose of testing. It now became relatively easy to transform a QTS to a QTA and then perform more quantitative testing with the QTA, without worrying whether the behavior of the system is influenced. This is of great importance when looking at the reason for this research: is a TA a reliable alternative for a QTS when testing? Therefore it is possible to do more practical testing with TAs when it comes to quiescence. With the transformations presented in this section it is made possible to easily create a TA as an alternative for the QTS and thus perform testing with quiescence in an easy and systematical way.

In Figure 7 it becomes clear that the transformation from the QTS to the QTA is indeed reversible by applying the description of Definition 11 to the QTA B from the figure. The QTS A from the figure is then created and therefore this example shows that the transformation is indeed reversible.

7. CONCLUSION

Section 3 up to section 6 answer research questions two up to five respectively. The research questions are defined in the introduction. First, in section 3 a way is provided to add quiescence to a timed automaton by defining the QTA. This section shows that all properties of the original TA stay intact when using QTAs. This is possible because an extra clock x is added to the TA which only purpose is to determine if quiescence is the case. In quiescent states this clock x makes sure that after a certain time t, quiescence is always made explicit by performing a transition with the special quiescence label δ. It is ensured this transition happens after a certain time t because of the invariant added to the quiescent state that $x \leq t$ and the clock constraint on the δ transition that x needs to be equal to t. In non-quiescent states the invariant that the clock x needs to be smaller or equal to the time t is also added to the state. Because of this invariant, it is possible in QTAs to determine whether there is an error in the system if no output or internal actions happens after a certain time t. The time t after which a state is marked quiescent is determined by the user.

In section 4 a way is presented to create a QTA from a labelled transition system. When creating QTAs from IOTSs two additions are needed: the extra clock x to determine if quiescence is the case and transitions with the special quiescence label δ to make quiescence explicit in quiescent states. This section showed that the original behavior and the original properties of the IOTS are not altered since the same input, output and internal transitions are possible. The IOTS is just extended and not changed. A small remark on this point is that with adding the invariant to non-quiescent states, it is possible that a state is too soon said to be quiescent and the system is erroneous when the time t chosen is too small. When this happens some input, output or internal actions can not occur anymore. In practice we assume that the user chooses the time t after which a state s is said to be quiescent in such a way that it does not alter the behavior of the system. Because of this, the behavior of the original IOTS is not altered and there are no limitations on this.

Section 5 describes to what extent QTSs as presented by Stokkink et al. [3] have the same behavior and structural properties as QTAs presented earlier in this paper. For this purpose we compare QTSs and QTAs created from the same IOTS. They are compared on two important things: trace equivalence and the well-formed rules. First, it is shown that the QTS and the QTA are trace equivalent since the QTA also contains the special transition for quiescence in quiescent states with the δ-label. Once again it is a issue here that the invariant in non-quiescent states can influence the transitions possible. Here it is also assumed that the user chooses the time t in such a way that this is not the case.This section also shows that if the well-formed rules defined by Stokkink et al. [3] apply on a QTS, they also apply on the QTA that is created from the same IOTS. There are no constraints on this. After this section it is therefore concluded that QTSs and QTAs can be used as alternatives for each other.

The last research question is answered in section 6. This section proposes a way to create a QTA from a QTS and vice versa. This is possible by adding or removing the clock x and the constraints on this clock respectively. In the section a proof is included that a QTS and a QTA created from this QTS are trace equivalent and that if the well-formed rules apply on the QTS, they also apply on the QTA. When creating a QTS from a QTA, this is also the case. Therefore it is possible to use a QTA instead of a QTS for practical testing when creating the QTA by Definition 10. Furthermore it becomes easier to actually use a QTA when you already have a QTS and want to save the trouble of using the original IOTS. It is even possible to create a QTA now without even having the original IOTS. The transformations of section 6 can therefore be a great stimulation (to start) usage of QTAs.

All sections described above together ensure that it is possible to model quiescence with timed automata and that this is a valid alternative for QTSs. Because in a QTA the modeling and determining of quiescence happens only by adding the extra clock x and the other properties of a TA are not influenced by this, this is a valid way to model quiescence with timed automata. Both a QTS and a QTA created from the same IOTS and a QTS and the QTA created from this QTS are trace equivalent, and in both cases it is ensured that if the well-formed rules apply on the QTS, they also apply on the QTA. This establishes that the QTS and the QTA can indeed be used as an alternative for each other. As mentioned multiple times before there is one remark on the usage of QTAs. The user needs to determine the time t after which is determined that quiescence is the case with QTAs. This time t depends heavily on the system and only the user can therefore determine this time t. The QTA only works as proposed if this time t is not too big or too small. When implementing QTAs this is something that always needs to be kept in mind and therefore it is also the biggest constraint on using QTAs. For the purpose of theoretical testing as presented in this paper, one can assume that the time t is chosen in such a way that quiescence is only detected where expected and the clock x does not influence any other behavior of the QTA then the detection of quiescence.

8. FUTURE WORK

As mentioned before this research only focuses on convergent and input-enabled transitions systems. For future work it will be very interesting to explore the possibilities of the QTA and its role as an alternative for a QTS when using divergent transition systems. A start for this is already made by Stokkink et al. [4] for QTSs. Another interesting topic for future work is to conduct more research in the time t after which quiescence is noticed. Now it is only left to the user to determine the time t and no standards are defined for this. Standardization may be possible after exploring more types of systems. This is definitely something that can influence usage of QTAs and it therefore seems like an important topic to explore. The last topic for future work will be the usage of more than one time t. In the QTA presented in this paper all the states are said to be quiescent after the same time t. A system that is modeled with the QTA can however have different behavior in different states and therefore the usage of more than one time t can make the determinisation of quiescence much more specific.

9. REFERENCES

[1] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science Volume 126, pp. 183-235*, 1992.

[2] J. Bengtsson and W. Yi. Timed automata: Semantics, algorithms and tools. *Lecture Notes in Computer Science Volume 3098, pp. 87-124*, 2004.

[3] G. Stokkink, M. Timmer, and M. Stoelinga. Talking quiescence: a rigorous theory that supports parallel composition, action hiding and determinisation. *EPCTS 80, pp. 73-87*, 2012.

[4] G. Stokkink, M. Timmer, and M. Stoelinga. Divergent quiescent transition systems. *Lecture Notes in Computer Science Volume 7942, pp. 214-231*, 2013.

[5] J. Tretmans. Test generation with inputs, outputs and repetitive quiescence, 1996.

[6] F. Vaandrager. A first introduction to uppaal.

[7] F. Vaandrager. On the relationship between process algebra and input/output automata. proc. *Proc. Of 6th Annual Symposium on Logic in Computer Science (LICS), IEEE, pp. 387-398*, 1991.