Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 51-7 (2018) 486–492

Minimizing Deterministic Timed Finite State Machines

Davide Bresolin * **Aleksandr Tvardovskii** **
Nina Yevtushenko *** **Tiziano Villa** **** **Maxim Gromov** **

* *University of Padova, Padova, Italy (e-mail: davide.bresolin@unipd.it).*
** *Tomsk State University, Tomsk, Russia (e-mail: tvardal@mail.ru, maxim.leo.gromov@gmail.com)*
*** *Institute for System Programming RAS, Moscow, Russia (e-mail: nyevtush@gmail.com)*
**** *University of Verona, Verona, Italy (e-mail: tiziano.villa@univr.it)*

Abstract: Timed automata and timed finite state machins (TFSMs) have been proposed to represent more accurately the behaviour of systems in continuous time. Recently, we introduced a model of TFSMs that extends the expressive power of FSMs by introducing a single clock, timed guards which restrict when the input/output transitions may happen, and timeouts on the transitions. We derived an abstraction procedure to convert a TFSM into an equivalent untimed FSM. Here, we extend the model with output timeouts and derive a minimal form for deterministic TFSMs that reduces the number of states, the number of transitions and the timeout values at each state.

Keywords: Timed Finite State Machines, Minimization, Real-time systems, Automata theory, Timed Automata

1. INTRODUCTION

Finite Automata (FA) and Finite State Machines (FSMs) are formal models of computation widely used in the theory and the practice of engineering and computer science, in application domains ranging from the study of computation and languages, sequential circuits design, communication protocols, embedded and reactive systems, software engineering, compilers, to biological modelling.

Finite Automata are language acceptors (also called recognizers or sequence detectors) and produce a binary output, indicating whether or not the received input is accepted. Extensions of the standard classes of FA with time constraints have been proposed since the 90s, to represent more accurately the behaviour of systems in real time. Timed Automata (TA) are such an example: they are finite automata augmented with a number of resettable real-time clocks, whose transitions are triggered by predicates over clock values (Alur and Dill, 1994). Timed automata include many variants (e.g., timed automata with a single clock that is reset at each transition Dima (2001)).

Finite state machines are language transducers that generate an output based on a given input and/or a state using actions, and are typically used to represent dynamical systems in control theory and in sequential circuits design. Timed models of FSMs are more recent and have been proposed in the literature by the introduction of time constraints such as timed guards or timeouts. Timed guards restrict the input/output transitions to happen within given time intervals. The meaning of timeouts is the following: if no input is applied at a current state for some timeout period, the timed FSM moves from the current state to another state using a timeout function. In particular, the timed FSMs proposed in Gromov et al. (2009) and El-Fakih et al. (2013, 2014) have the following features: one clock variable, time constraints to limit the time elapsed at a state, and clock reset when a transition is executed. The timed FSMs proposed in Merayo et al. (2008) and Hierons et al. (2009) have the following features: one clock variable, time constraints to limit the time elapsed when an output has to be produced after an input has been applied to the FSM, clock reset when an output is produced, timeouts.

In Bresolin et al. (2014) we proposed a timed FSM model with a single clock that includes both timed guards and timeouts, compared its expressive power with previous TFSM models and derived an abstraction procedure to convert a TFSM into an equivalent untimed FSM. In this paper, we continue our investigation on such a TFSM model by extending it with output timeouts and by deriving a minimal form for deterministic TFSM that reduces the number of states, the number of transitions and the timeout values at each state. Minimality is a useful property in applications, e.g., methods for test derivation usually take a minimal specification as an input, otherwise tests become much longer. Preliminary results were published in a reference in Russian (Tvardovskii et al., 2017).

2. REDUCTION OF TIMED TRANSITION SYSTEMS

Since the 90s there have been investigations on the minimization of timed automata, meaning the reduction of their resources in the form of states, transitions, clocks and constants used in guards. State minimization has led

10.1016/j.ifacol.2018.06.344

to defining regions of equivalent states by the introduction of quotient operations with respect to bisimulation equivalence (Alur et al., 1992; Yannakakis and Lee, 1997). Springintveld and Vaandrager (1996) introduced a class of automata called minimizable whose clocks have bounded time domains, from which minimal ones can be derived by bisimulation. The removal of inactive clocks (in a discrete state they are reset before being tested) has been considered by C. Daws (1996). Computing the minimum number of clocks or the smallest guard constants has been shown to be undecidable by Tripakis (2006). In the same paper, it is mentioned as an open problem the minimization of the number of discrete states, and the trade-off between the discrete states vs. an increase of the number of clocks or of the size of the constants.

In the context of discrete event system specification (DEVS), a formalism called Timed Sequential Machines (TSMs) was introduced by Giambiasi (2014). TSMs have stable and transitory states, where the latter have a finite lifetime at whose expiration an internal transition with output emission takes place, if no external event occured before the lifetime expiration; stable states have an infinite lifetime and react only to external events by transiting to the next state. Methods were proposed for the minimization of finite-state completely and incompletely specified TSMs, extending the standard algorithms of the literature for completely and incompletely specified FSMs (Rho et al., 1994; Kam et al., 1997).

3. PRELIMINARIES

In this section we introduce the definitions and notations used throughout the paper. Given a finite alphabet A, a *timed symbol* is a pair (a, t) where $t \in \mathbb{R}$ is called the *delay* of the symbol $a \in A$. A timed sequence is then defined as a finite sequence $(a_1, t_1)(a_2, t_2)(a_3, t_3)\dots$ of timed symbols where the delays t_i are non-negative. An FSM with timed guards and timeouts (TFSM) is an FSM annotated with a *clock* that operates by reading a *timed input sequence* $(i_1, t_1)(i_2, t_2)\dots(i_k, t_k)$ defined on some *input alphabet* I, and producing a corresponding *timed output sequence* $(o_1, t_1)\ (o_2, t_2)\dots(o_k, t_k)$ on some *output alphabet* O. The clock is a real number that measures the time delay at a state, and its value is reset to zero when a transition is executed. Our TFSM model has input and output timeouts and input timed guards, and extends the ones in Bresolin et al. (2014) with the addition of output timeouts. When an input timeout expires at state s, the TFSM spontaneously moves to another state. A good example is a mobile phone when the screen becomes dark if there is no touch or pressed button for a number of time units, i.e., no input is applied. Input timed guards describe the behavior at a given state for inputs which arrive at different time instants. An output timeout describes how long an applied input is processed at a given state before the related output is emitted. Correspondingly, a TFSM is a 5-tuple $\mathsf{S} = (I, S, O, \lambda_S, \Delta_S)$ where I and O are input and output alphabets, S is a finite non-empty set of states, $\lambda_S \subseteq (S \times I \times O \times S \times \Pi \times Z)$ is the *transition relation* (*behavior*) and Δ_S is the timeout function. The set Π is a set of *input timed guards*, and Z is a set of *output delays* which are nonnegative integers. The *timeout function* is a function $\Delta_S : S \mapsto S \times (N \cup \{\infty\})$ where N is the set of

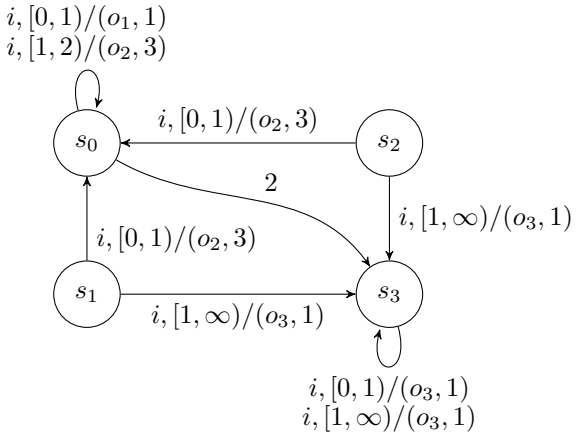

Fig. 1. Example of a TFSM with timed guards, timeouts and output delays.

nonnegative integers: for each state this function specifies the maximum time to wait for an input. An input timed guard $g \in \Pi$ describes the time domain when a transition can be executed and is given in the form of a bounded interval $\lceil min, max \rceil$ from $[0, T)$, where $\lceil \in \{(, [\}$, $\rceil \in \{),]\}$ and T is the value of the (input) timeout at the current state. An output delay defines how long does it take to produce an output after applying an input. The transition $(s, i, o, s', g, d) \in S \times I \times O \times S \times \Pi \times Z$ means that the TFSM S being at state s accepts an input i applied at time $t \in g$ measured from the moment when TFSM S entered state s. Upon reception of i, the clock is set to zero and S produces output o after d time units counted from the moment when the input has been applied; the clock is then reset to zero to start measuring the delay of the next timed input. Given state s of a TFSM S such that $\Delta_S(s) = (s', T)$, if no input is applied before the timeout T expires, the TFSM S moves to state s' and the clock is set to zero.

We would like to emphasize that in our definition TFSMs are *non-initialized*: there is no notion of initial state, and the execution of the machine can start from any state in S. In the following, we also briefly consider *initialized* TFSMs equipped with an *initial state* $s_0 \in S$: in this case the execution of the machine should start from state s_0.

Figure 1 shows an example of an unitialized TFSM with four states, one input symbol i and three output symbols o_1, o_2, o_3. The label "2" on the transition from s_0 to s_3 represents a timeout $\Delta_S(s_0) = (s_3, 2)$ for the state s_0. For all the other states the value of the timeout is ∞ (i.e., no timeouts).

A TFSM is *complete* if for each state s and timed input (i, t) there exists at least one transition (s, i, o, s', g, d) such that $t \in g$, and it is *deterministic* if there exists at most one such transition. Hence, in a complete and deterministic TFSM there exists exactly one transition that can be activated for every input and time instant. Given two complete and deterministic TFSMs S and P and their states s and p, states s and p are *equivalent* if the output responses at these states coincide for each timed input sequence; otherwise, the states are *distinguishable*. TFSM S is *state-reduced* if the states of the TFSM are pairwise distinguishable. Two non-initialized complete de-

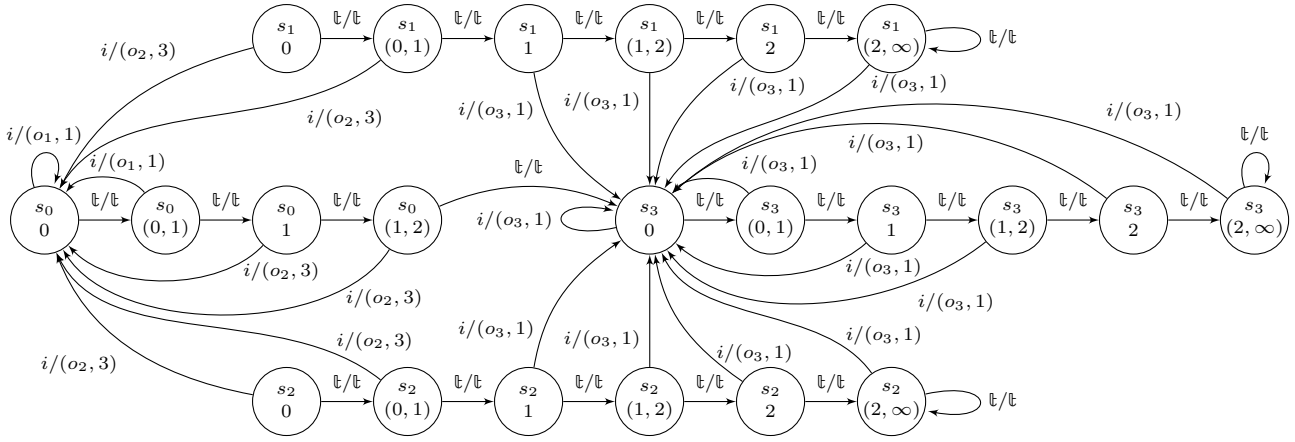

Fig. 2. FSM abstraction of the TFSM of Figure 1.

terministic TFSMs S and P are *equivalent* if for each state of TFSM S there exists an equivalent state of TFSM P, and vice versa. Two initialized complete deterministic TFSMs S and P are *equivalent* if their initial states are equivalent.

We are aware that strictly speaking non-initialized TFSMs are non-deterministic because of the plurality of initial states (so that the same input sequence may generate different output sequences, according to the initial state), but here we call them as deterministic if for a given initial state and input sequence there is at most one output sequence.

Two TFSMs S and P are *isomorphic* if there exists a one-to-one correspondence $H : S \mapsto P$ such that there exists a transition $(s, i, o, s', g, d) \in \lambda_S$ if and only if there exists a transition $(H(s), i, o, H(s'), g, d) \in \lambda_S$ and $\Delta_P(H(s)) = (H(s'), T)$ if $\Delta_S(s) = (s', T)$.

Finally, an FSM is a 4-tuple $\mathsf{S} = (I, S, O, \lambda_S)$ where I and O are input and output alphabets, S is a finite non-empty set of states, $\lambda_S \subseteq (S \times I \times O \times S)$ is the *transition relation* (*behavior*).

4. DERIVING THE FSM ABSTRACTION OF TFSM

The behavior of a TFSM can be adequately described using a standard FSM that is called the *FSM abstraction* of the TFSM and that is derived by extending the result in Bresolin et al. (2014) to consider output delays as follows.

Given a complete deterministic TFSM $\mathsf{S} = (S, I, O, \lambda_S, \Delta_S)$, and the largest finite bound of input timed guards B and maximum output delay D, we derive the FSM abstraction of TFSM S as the FSM $FSM(\mathsf{S}) = (S_{FSM}, I \cup \{\mathbb{t}\}, O_{FSM}, \lambda_{FSM})$ where $S_{FSM} = \{(s, 0), (s, (0, 1)), (s, 1), (s, (1, 2)), \ldots, (s, B-1), (s, (B-1, B)), (s, B), (s, (B, \infty)) : s \in S\}$, $O_{FSM} = \{(o, 0), (o, 1), \ldots, (o, D) : o \in O \cup \{\mathbb{t}\}\}$. The input $\mathbb{t}$ is a special input of the FSM abstraction. Given states (s, t), $t = 0, \ldots, B$, of FSM $FSM(\mathsf{S})$ and input i, a transition $((s, t), i, (o, d), (s', 0))$ is a transition of the FSM abstraction $FSM(\mathsf{S})$ if and only if there exists a transition $(s, i, o, s', g, d) \in \lambda_S$ such that $t \in g$. Given states (s, h), $h = (0, 1), \ldots, (B-1, B), (B, \infty)$, of FSM $FSM(\mathsf{S})$ and input i, a transition $((s, h), i, (o, d), (s', 0))$ is a transition of $FSM(\mathsf{S})$ if and only if there exists a transition $(s, i, o, s', g, d) \in \lambda_S$ such that $h \subseteq g$. In other words, transitions under input $i \in I$ correspond to timed inputs (i, t) where t is 'hidden' as the second item of states of the FSM abstraction $FSM(\mathsf{S})$. Transitions under the special input $\mathbb{t}$ correspond to the clock change between non-integer and integer values, or to a timeout transition between states. Given state s such that $\Delta_S(s) = (s', T)$, transitions $((s, n), \mathbb{t}, \mathbb{t}, (s, (n, n+1)))$ and $((s, (n-1, n)), \mathbb{t}, \mathbb{t}, (s, n))$ are in the transition relation λ_{FSM} if and only if $n < T$; transition $((s, (n-1, n)), \mathbb{t}, \mathbb{t}, (s', 0)) \in \lambda_{FSM}$ if and only if $n = T < \infty$. In Bresolin et al. (2014) it is shown that the FSM abstraction of complete and deterministic TFSM S is also complete and deterministic.

A timed input sequence α of TFSM S can be transformed into a corresponding input sequence α_{FSM} of the FSM abstraction $FSM(\mathsf{S})$. In this case, each timed input (i, t) is replaced by sequence $\mathbb{t}, \mathbb{t}, \ldots, \mathbb{t}, i$, of inputs of the FSM abstraction where the number of inputs $\mathbb{t}$ equals the number of clock transitions between a non-integer and an integer value for the time duration t. At the same time the response of the FSM abstraction to sequence $\mathbb{t}, \mathbb{t}, \ldots, \mathbb{t}, i$ equals $\mathbb{t}, \mathbb{t}, \ldots, \mathbb{t}, (o, d)$, where the number of inputs $\mathbb{t}$ is the same as for the timed input (i, t) and (o, d) is the response of the TFSM to timed input (i, t). Thus, the output sequence of the FSM abstraction γ_{FSM} can be transformed into a corresponding timed output sequence γ by removing all outputs $\mathbb{t}$. A pair that associates a timed input sequence α with the corresponding timed output senquence γ is called a *timed trace*. Using the results in Bresolin et al. (2014) the following statement can be established.

Proposition 1. A timed trace α/γ exists for TFSM S if and only if there exists a trace $\alpha_{FSM}/\gamma_{FSM}$ for the FSM abstraction $FSM(\mathsf{S})$.

Figure 2 shows the FSM abstraction of the TFSM S from Figure 1, obtained by applying the above procedure. In this case the largest finite bound of input timed guards is $B = 2$, and thus the states of the abstraction are of the form $(s_i, 0), (s_i, (0, 1)), \ldots, (s_i, 2), (s_i, (2, \infty))$ for $i = 0, 1, 2, 3$. The transition from state $(s_0, (1, 2))$ to state $(s_3, 0)$ labelled with $\mathbb{t}/\mathbb{t}$ represents the timeout $\Delta_S(s_0) = (s_3, 2)$ of S: when the value of the clock is equal to 2 the machine is forced to move to state s_3. As a consequence, state $(s_0, (2, \infty))$ is not included in the abstraction. Conversely, the self-loops labelled with $\mathbb{t}/\mathbb{t}$ at states $(s_1, (2, \infty))$, $(s_2, (2, \infty))$ and $(s_3, (2, \infty))$ repre-

sent the fact that states s_1, s_2 and s_3 have no timeout in the original TFSM. Transitions labelled with actual input/output symbols start from states where the second component is compatible with the timed guard in the original timed transitions of S and end into states where the second component is always 0, to represent the fact that in a TFSM the clock is reset at every transition.

According to Proposition 1, all the trace features of a TFSM are preserved for its FSM abstraction and thus, the state equivalence of a TFSM can be analyzed based on a standard FSM. However, states of this FSM abstraction have the information about time properties which should be taken into account when minimizing a TFSM. The following statement establishes necessary and sufficient conditions for two TFSM states to be equivalent.

Proposition 2. States s_1 and s_2 of TFSM S are equivalent if and only if states $(s_1, 0)$ and $(s_2, 0)$ of the FSM abstraction $FSM(\mathsf{S})$ are equivalent.

In fact, states s_1 and s_2 of TFSM S are equivalent if and only if for each timed input sequence, the output sequences at these states coincide. By Proposition 1, the latter means that for each input sequence, the output responses at states $(s_1, 0)$ and $(s_2, 0)$ of $FSM(\mathsf{S})$ also coincide. Thus, the conclusion about the state equivalence of a given TFSM can be drawn based on its FSM abstraction using the usual algorithms for checking FSM equivalence (Gill, 1962).

Our abstraction procedure may be compared with time-abstracting bisimulations defined for timed automata, which yield - by a quotient operation - a finite region graph preserving satisfiability of formulas written in timed computation tree logic (TCTL, see Tripakis and Yovine (1996), Yovine (1998) and Tripakis and Yovine (2001)). Our specific algorithm applies to transducers with a single clock and preserves also output observational equivalence (besides clock equivalence).

5. DERIVING MINIMAL FORM OF TFSM

According to Proposition 2, a procedure for deriving a state-reduced form of a TFSM is then reduced to derive the FSM abstraction $FSM(\mathsf{S})$ of TFSM S and the partition E_{FSM} into equivalent states of the abstraction $FSM(\mathsf{S})$. When the original TFSM is complete and deterministic, by construction, the corresponding FSM abstraction is also complete and deterministic and thus the general procedure for deriving the partition of an FSM into equivalent states can be applied for minimizing $FSM(\mathsf{S})$. In the second step of the procedure the partition E into equivalent states of TFSM S is derived: states s_1 and s_2 of TFSM S are in the same class of E if and only if states $(s_1, 0)$ and $(s_2, 0)$ are in the same class of E_{FSM}. A state-reduced form of S is derived using the partition E in the usual way. For example, after applying the above procedure to TFSM S in Figure 1, the TFSM in Figure 3 is obtained and this TFSM is state-reduced and equivalent to S, i.e., it is a state-reduced form of TFSM S.

All state-reduced forms of a TFSM S have the same number of states, but differently from complete deterministic FSMs, in general, a state-reduced form of a TFSM is not unique. The reason is that for timed FSMs not only the state set should be minimized as it happens for standard

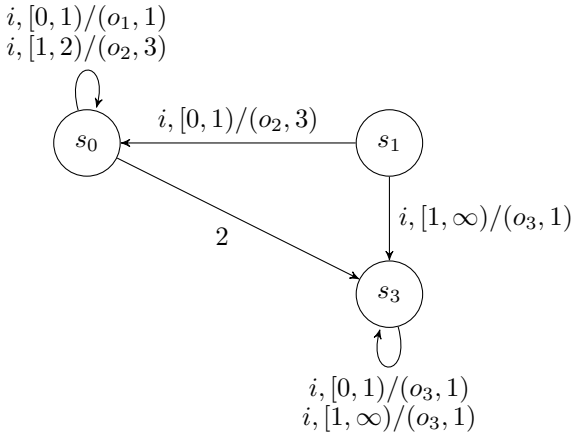

Fig. 3. A state-reduced form of TFSM S in Figure 1.

FSMs but also timed guards and timeouts need to be minimized too. In particular, in some cases, input timed guards at some states could be merged as well as the value of input timeouts could be minimized. For example, in the following we will prove, by applying our minimization procedure, that the timeout $\Delta_S(s_0) = (s_3, 2)$ of the state-reduced TFSM S in Figure 3 can be replaced with timeout $\Delta_S(s_0) = (s_1, 1)$ without breaking the equivalence relation between TFSMs in Figures 1 and 3. In Tvardovskii et al. (2017), a class of time-reduced TFSMs for which time aspects are also optimized is introduced.

An FSM with timed guards and timeouts S is *time-reduced* if for each two transitions $(s, i, o, s', g_1, d), (s, i, o, s', g_2, d) \in \lambda_S$ it holds that timed guards g_1 and g_2 cannot be merged into a single guard, namely, that $g_1 \cup g_2$ is not an interval of the form $\lceil min, max \rceil$. Moreover, for each state s such that $\Delta_S(s) = (s', T)$, it holds that for each state s'' and integer $T' < T$, TFSM S' which is obtained from S by replacing the timeout at state s by $\Delta_S(s) = (s'', T')$, is not equivalent to S. Given a deterministic complete TFSM S, a state- and time-reduced TFSM P that is equivalent to S is a *minimal form* of TFSM S.

If for two transitions $(s, i, o, s', g_1, d), (s, i, o, s', g_2, d) \in \lambda_S$ of TFSM S timed guards g_1 and g_2 can be merged into a single guard, then these two transitions can be replaced by a transition $(s, i, o, s', g_1 \cup g_2, d)$. Minimal timeouts for states of TFSM S can be found by analyzing the FSM abstraction $FSM(\mathsf{S})$. In order to find a minimal timeout at state s, states $(s, 1), \ldots, (s, j), \ldots$ of the FSM abstraction are considered. If there exists a state $(s', 0)$ such that states $(s', 0)$ and (s, j) are equivalent, then a transition $((s, (j-1, j)), \mathbb{t}, \mathbb{t}, (s, j))$ of $FSM(\mathsf{S})$ can be replaced by a transition $((s, (j-1, j)), \mathbb{t}, \mathbb{t}, (s', 0))$. This means that the timeout at state s of TFSM S can be replaced by $\Delta_S(s) = (s', j)$ without changing the TFSM behavior. Thus, for each state s, a minimal timeout that does not change the behavior of the TFSM can be found.

Hence, given a TFSM S, a minimal form of S, i.e., a state-time-reduced TFSM, can be derived using two operations: (1) transitions under the same input where timed guards can be merged should be replaced by a single transition; (2) each timeout should be set to the minimum value.

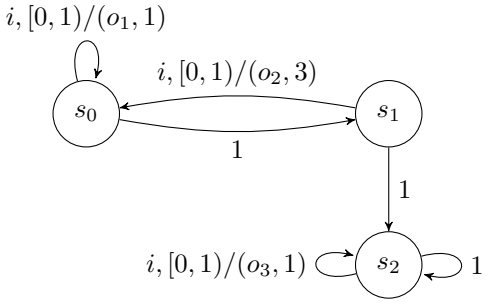

Fig. 4. State- and time-reduced form of TFSM S (Figure 1).

The following procedure derives such a state- and time-reduced TFSM.

(1) Replace every pair of transitions (s, i, o, s', g_1, d), (s, i, o, s', g_2, d) of S such that g_1 and g_2 can be merged into a single timed guard, by a transition $(s, i, o, s', g_1 \cup g_2, d)$.
(2) Derive the FSM abstraction $FSM(\mathsf{S})$ of TFSM S and the partition E_{FSM} into equivalent states of the abstraction $FSM(\mathsf{S})$. Merge every pair of states s_1, s_2 of S such that $(s_1, 0)$ and $(s_2, 0)$ are equivalent.
(3) For each state s of TFSM S where $\Delta_S(s) = (s', T)$, with $T \neq \infty$, determine whether there exists state (s, j) of $FSM(\mathsf{S})$, $1 \leq j \leq T-1$, with minimal j such that there exists state $(s'', 0)$ which is equivalent to (S, j). If there exists such pair of states (s, j) and $(s'', 0)$ then replace timeout at state s to $\Delta_S(s) = (s'', j)$.
(4) For each state s of TFSM S where $\Delta_S(s) = (s', \infty)$, determine whether there exists a state (s, j) of $FSM(\mathsf{S})$, $1 \leq j \leq B$, with minimal j such that there exists state $(s'', 0)$ which is equivalent to (s, j). If there exists such pair of states (s, j) and $(s'', 0)$ then replace timeout at state s by $\Delta_S(s) = (s'', j)$.
(5) For each state s of TFSM S where $\Delta_S(s) = (s', \infty)$, which has not been modified at the previous step, determine whether there exists a state $(s'', 0)$ of $FSM(\mathsf{S})$ such that states $(s'', 0)$ and $(s, (B, \infty))$ are equivalent. If there exists such a state $(s'', 0)$, then replace timeout at state s by $\Delta_S(s) = (s'', B+1)$.
(6) For each state s of TFSM S where $\Delta_S(s) = (s', T)$ and $T < \infty$, remove any transition (s, i, o, s', g, d) such that $g \cap [0, T) = \emptyset$. If for a transition $(s, i, o, s', \langle A, B\rangle, d)$ it holds that $\langle A, B\rangle \cap [0, T) \neq \emptyset$ but $\langle A, B\rangle$ is not contained in $[0, T)$, then replace the transition $(s, i, o, s', \langle A, B\rangle, d)$ by a transition $(s, i, o, s', \langle A, T\rangle, d)$.

Proposition 3. Given a complete deterministic TFSM S, the above procedure returns a state- and time-reduced TFSM P that is equivalent to S.

Proof. Step 1 where transitions (s, i, o, s', g_1, d) and (s, i, o, s', g_2, d) are merged as a transition $(s, i, o, s', g_1 \cup g_2, d)$, does not change the behavior of TFSM S.

Step 2 builds the partition E_{FSM} of equivalent states and then merges equivalent states. By Proposition 2, this operation does not change the behaviour of S.

Consider Step 3 of the procedure for replacing timeouts. Let s and s'' be two states of TFSM S such that $\Delta_S(s) = (s', T)$, and suppose there exists a minimal $j < T$, such that states (s, j) and $(s'', 0)$ of the FSM abstraction $FSM(\mathsf{S})$ are equivalent. The pair of states (s, j) and $(s'', 0)$ will be identified at Step 2 of the procedure. By Proposition 2, if states (s, j) and $(s'', 0)$ are equivalent then the output response of TFSM S at state s'' to each timed input sequence coincides with the corresponding response of TFSM S at state s at time instant j. Thus, the replacement of a timeout $\Delta_S(s) = (s', T)$ at state s by $\Delta_S(s) = (s'', j)$ does not change the behavior of TFSM S at state s. Similarly, the replacement of a timeout at Steps 4 and 5 of the procedure does not change the behavior of TFSM S.

At Step 6 of the procedure, a transition (s, i, o, s', g, d) is removed if the input timed guard g and $[0, T)$ are disjoint where $\Delta_S(s) = (s'', T)$, since this transition cannot be executed in the TFSM S. Transition (s, i, o, s', g, d) such that input timed interval $g = [A, B)$ intersects $[0, T)$ but the upper bound of g is beyond the value of timeout T cannot be executed in the TFSM S for the interval $[T, B)$. Thus, removing and bounding a transition at Step 6 does not change the behavior of TFSM S.

We now show that TFSM P is time-reduced. Assume that for state p_0 of TFSM P the timeout $\Delta_P(p_0) = (p_1, T_1)$ is replaced by a timeout $\Delta_P(p_0) = (p_2, T_2)$ where $T_1 > T_2$, and the behavior of TFSM P at state p_0 is not changed. The latter means that in the FSM abstraction $FSM(\mathsf{P})$ there exists either a pair of equivalent states (p_0, T_2) and $(p_2, 0)$ or a pair of equivalent states $(p_0, (T_2 - 1, \infty))$ and $(p_2, 0)$. The pair of equivalent states (p_0, T_2) and $(p_2, 0)$ does not exist in $FSM(\mathsf{P})$ (Steps 3 and 4 of the procedure). If there exist equivalent states $(p_0, (T_2 - 1, \infty))$ and $(p_2, 0)$ in $FSM(\mathsf{P})$ then the output response of TFSM P at state p_2 to each timed input sequence coincides with that of TFSM P at state p_0 starting from the time instant $T > (T_2 - 1)$. In this case, by construction of the FSM abstraction $FSM(\mathsf{P})$ has no state (p_0, T_2), but according to Step 5 of the procedure, the timeout at state p_0 is replaced by $\Delta_P(p_0) = (p_2, T_2)$. Thus, equivalent states $(p_0, (T_2 - 1, \infty))$ and $(p_2, 0)$ do not exist in $FSM(\mathsf{P})$. □

As an example, the minimal form of TFSM S (Figure 1) obtained by the above procedure is shown in Figure 4. By direct inspection, one can assure that in Figure 4 the timeout $\Delta_S(s_0) = (s_2, 2)$ has been replaced by the timeout $\Delta_S(s_0) = (s_1, 1)$, while a transition $(s_0, i, o_2, s_0, [1, 2), 3)$ has been removed. At the same time, two transitions from state s_2 have been merged and timeouts $\Delta_S(s_1) = (s_1, \infty)$ and $\Delta_S(s_2) = (s_2, \infty)$ were replaced by $\Delta_S(s_1) = (s_2, 1)$ and $\Delta_S(s_2) = (s_2, 1)$, respectively.

The above procedure can be used to obtain state- and time-reduced TFSMs for both initialized and non-initialized machines. However, for initialized TFSMs the result is not necessarily unique up to equivalence. Figure 5 shows an example of two state- and time-reduced TFSMs that are equivalent but not isomorphic. Notice that when considered as non-initialized TFSMs the two machines are not equivalent: in particular, the TFSM P can generate output o_2 only when the input i is received with a delay of more that 2 time units, while TFSM Q can generate

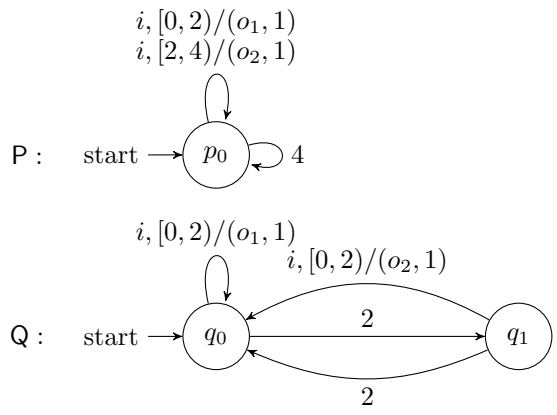

Fig. 5. Example of two equivalent state- and time-reduced *initialized* TFSMs.

output o_2 also when the delay is less than 2 time units by starting from state q_1. In the following we prove that for non-initialized TFSMs the minimal form is indeed unique.

Theorem 4. Two non-initialized deterministic complete state- and time-reduced TFSMs are equivalent if and only if they are isomorphic.

Proof. $\Rightarrow$ Consider two deterministic complete state- and time- reduced TFSMs S and P that are equivalent. Since we are considering non-initialized, complete and deterministic machines, the two machines are equivalent if and only if for every state s of S we can find an equivalent state p of P, and vice-versa. Hence, since both TFSMs are reduced they must have the same number of states. Consider the one-to-one correspondence $H : S \mapsto P$ such that $H(s)$ is a state of TFSM P which is equivalent to state s. We now show that for each pair of states s and $p = H(s)$ such that $\Delta_S(s) = (s', T_s)$ and $\Delta_S(p) = (p', T_p)$, it holds that $T_s = T_p$ and $p' = H(s')$. If $T_p < T_s$ then due to the fact that S and P are equivalent, there exists state s'' which is equivalent to state p'. Since states s and p are also equivalent, the timeout at state s can be replaced by $\Delta_S(s) = (s'', T'_s)$ where $T'_s = T_p < T_s$. The latter is not possible as S is time-reduced. Since P is also time-reduced, the same reasoning applies when $T_s < T_p$. Thus, $T_p = T_s$. Since states s and p are equivalent, states s' and p' are also equivalent and respectively, $p' = H(s')$. Similar to Tvardovskii and Yevtushenko (2014), since TFSMs S and P are equivalent, there exists a transition $(s, i, o, s', g, d) \in \lambda_S$ if and only if there exists a transition $(H(s), i, o, H(s'), g, d) \in \lambda_P$. Thus, TFSMs S and P are isomorphic.

$\Leftarrow$ Since isomorphic TFSMs coincide up to state renaming, isomorphic TFSMs are equivalent. $\square$

Corollary 5. Given a non-initialized deterministic complete TFSM S, two state- and time-reduced forms of TFSM S are isomorphic.

Theorem 6. Given a deterministic complete TFSM S, the minimal form of S is unique.

According to Theorem 6, for FSMs with timed guards and timeouts, there exists the minimal (canonical) form that is a well-defined state- and time-reduced TFSM. As a corollary, similar statements can be drawn for FSMs only with timed guards or only with timeouts.

Corollary 7. Given a non-initialized deterministic complete FSM S only with timed guards (or only with timeouts), the minimal form of S is unique.

6. CONCLUSIONS

In this paper we considered a TFSM model with a single clock that includes timed guards, timeouts and output delays. Then we derived a procedure to build a minimal form for deterministic TFSMs that reduces the number of states, the number of transitions and the timeout values at each state, and it is unique up to isomorphism for non-initialized TFSMs.

We believe that our model of timed FSMs strikes a good balance between expressive power and ease of analysis, and we plan to continue the research along different lines. A first line is the extension of the minimization procedure to nondeterministic TFSMs. We conjecture that this could be done by replacing the notion of equivalent states with a weaker one, which can be computed for nondeterministic machines, like bisimulation, and then by using a procedure similar to the one proposed in this paper to obtain a state- and time-reduced TFSM. In this case, however, it would be very difficult, if not impossible, to guarantee uniqueness of the obtained minimal machine, which does not hold even for non-deterministic finite automata (see Kameda and Weiner (1970)).

A second line of extension is to study the composition of TFSMs to find sufficient conditions that guarantee that the composed system can be represented by a single-clock TFSM, and to derive algorithms that compute the composition. When considering untimed FSMs, a slow environment and the absence of livelocks (i.e., states in a loop making no progress, which means absence of combinational cycles in the standard composition of FSMs) is sufficient to guarantee that the composition is a complete deterministic FSM. However, it is not the case for TFSMs: we have some preliminary examples where the composition of two single-clock TFSMs cannot be represented using only one clock. This suggests that limitations on the communication between the components and the external environment must be introduced to obtain composition operators that are closed with respect to single-clock TFSMs.

Future work includes also deriving tests for TFSMs with timed guards, timeouts and output delays, and investigating the solution of equations over timed FSMs to solve control synthesis problems.

ACKNOWLEDGEMENTS

Maxim Gromov, Alexander Tvardovskii, Nina Yevtushenko were partially funded by the Russian Science Foundation (RSF), Project # 16-49-03012. Tiziano Villa was partially supported by MIUR, Project "Italian Outstanding Departments, 2018-2022". Davide Bresolin and Tiziano Villa acknowledge partial support by INDAM, GNCS 2018, "Formal Methods for the Verification and Synthesis of Discrete and Hybrid Systems".

REFERENCES

Alur, R., Courcoubetis, C., Halbwachs, N., Dill, D., and Wong-Toi, H. (1992). Minimization of timed transition

systems. In W. Cleaveland (ed.), *CONCUR '92: Third International Conference on Concurrency Theory Stony Brook, NY, USA, August 24–27, 1992 Proceedings*, 340–354. Springer Berlin Heidelberg, Berlin, Heidelberg.

Alur, R. and Dill, D.L. (1994). A theory of timed automata. *Theoretical Computer Science*, 126(2), 183 – 235.

Bresolin, D., El-Fakih, K., Villa, T., and Yevtushenko, N. (2014). Deterministic timed finite state machines: Equivalence checking and expressive power. In *Proc. of the 5th International Symposium on Games, Automata, Logics and Formal Verification (GandALF 2014)*, volume 161 of *EPTCS*, 203–216. Open Publishing Association.

C. Daws, S.Y. (1996). Reducing the number of clock variables of timed automata. In *Proc. of the 17th IEEE Real-Time Systems Symposium, RTSS 1996*.

Dima, C. (2001). Real-time automata. *J. Autom. Lang. Comb.*, 6(1), 3–23.

El-Fakih, K., Gromov, M., Shabaldina, N., and Yevtushenko, N. (2013). Distinguishing experiments for timed nondeterministic finite state machines. *Acta Cybernetica*, 212(2), 205–222.

El-Fakih, K., Yevtushenko, N., and Simao, A. (2014). A practical approach for testing timed deterministic finite state machines with single clock. *Science of Computer Programming*, 80, Part B(0), 343 – 355.

Giambiasi, N. (2014). An introduction to timed sequential machines. *Simulation: Transactions of the Society for Modeling and Simulation International*, 90(3), 337–352.

Gill, A. (1962). *Introduction to the theory of finite state machines*. McGraw-Hill.

Gromov, M., El-Fakih, K., Shabaldina, N., and Yevtushenko, N. (2009). Distinguishing non-deterministic timed finite state machines. In *Formal Techniques for Distributed Systems*, volume 5522 of *Lecture Notes in Computer Science*, 137–151. Springer Berlin Heidelberg.

Hierons, R.M., Merayo, M.G., and Núñez, M. (2009). Testing from a stochastic timed system with a fault model. *The Journal of Logic and Algebraic Programming*, 78(2), 98 – 115.

Kam, T., Villa, T., Brayton, R., and Sangiovanni-Vincentelli, A. (1997). *Synthesis of FSMs: functional optimization*. Kluwer Academic Publishers, Boston.

Kameda, T. and Weiner, P. (1970). On the state minimization of nondeterministic finite automata. *IEEE Transactions on Computers*, C-19(7), 617–627.

Merayo, M.G., Núñez, M., and Rodríguez, I. (2008). Formal testing from timed finite state machines. *Computer Networks*, 52(2), 432–460.

Rho, J., Hachtel, G.D., Somenzi, F., and Jacoby, R.M. (1994). Exact and heuristic algorithms for the minimization of incompletely specified state machines. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 13(2), 167–177.

Springintveld, J. and Vaandrager, F. (1996). Minimizable timed automata. In B. Jonsson and J. Parrow (eds.), *Formal Techniques in Real-Time and Fault-Tolerant Systems: 4th International Symposium Uppsala, Sweden, September 9–13, 1996 Proceedings*, 130–147. Springer Berlin Heidelberg, Berlin, Heidelberg.

Tripakis, S. (2006). Folk theorems on the determinization and minimization of timed automata. *Information Processing Letters*, 99(6), 222 – 226.

Tripakis, S. and Yovine, S. (1996). Analysis of timed systems based on time-abstracting busimulations. In *8th International Conference on Computer-Aided Verification (CAV)*, volume 1102 of *Lecture Notes in Computer Science*, 232–243. Springer Berlin Heidelberg.

Tripakis, S. and Yovine, S. (2001). Analysis of timed systems based on time-abstracting busimulations. *Formal Methods in System Drsign*, 18(1), 25–68.

Tvardovskii, A.S. and Yevtushenko, N.V. (2014). Minimizing FSMs with timed guards. In *Vestnik TGU.*, volume 29, issue 2 of *Upravlenie, vychislitelnaya technika i informatica*, 77–83. In Russian.

Tvardovskii, A.S., Yevtushenko, N.V., and Gromov, M.L. (2017). Minimizing finite state machines with timed guards and timeouts. In *Trudy ISP RAN/Proc. ISP RAS*, volume 29, issue 4 of *Proceedings of the Institute for System Programming*, 139–154. In Russian.

Yannakakis, M. and Lee, D. (1997). An efficient algorithm for minimizing real-time transition systems. *Formal Methods in System Design*, 11(2), 113–136.

Yovine, S. (1998). Model checking timed automata. In G. Rozenberg and F.W. Vaandrager (eds.), *Lectures on Embedded Systems: European Educational Forum School on Embedded Systems Veldhoven, The Netherlands November 25–29, 1996*, 114–152. Springer Berlin Heidelberg, Berlin, Heidelberg.