# HIERARCHICAL MODELS OF TIMED CONCURRENT SYSTEMS: COMPOSITIONAL SEMANTICS

**Jaroslav Fogel**
**Institute of Informatics Slovak Academy of Sciences, Bratislava**

**Abstract**
In the paper, we present the timed extension of the hierarchical state machines with the compositional semantics ensuring that the semantics of the model can be determined from the semantics of its components. We proposed the compositional semantics of traces with the goal of its successive utilization in the analysis and verification of the model properties. At the end of the paper, we discussed the computational complexity of the reachability problem, and its decomposition to sub-problems with the lower complexity as the consequence of the compositional semantics and state refinement.

**Keywords:** Hierarchical state machine, timed models, compositional semantics, verification, state reachability.

## 1. INTRODUCTION

Large industrial processes require models, which inherently express their hierarchical structure, concurrency and dynamics. Mainly, dynamics requires introducing the timing aspects into the model that is important step in the construction of real- time control systems for such processes. As a whole, the model of the control system (which can be done automatically) and the user- supplied model of the controlled plant can be used for the verification of the controller properties concerning mainly the real-time requirements. The specification problem for these real-time applications is more complex since the absolute timing behaviour and not only the functional behaviour of a system is important.
The paper describes a specification language based on the hierarchical state machines (HSM) extended with timing aspects called timed HSM (THSM). We give the compositional semantics of THSM ensuring that the semantics of THSM can be determined from the semantics of its components. Compositionality is also useful when formally analysing the reachability problem.
There are several works dealing with HSM as e.g., [1], [2] where authors describe finite state machines extended with both the hierarchy and concurrency but without timing aspects. Many works is devoted to the theory of timed automata [3], [4], from which we borrow the basic syntax concerning the clocks (time variables). Similar work is done in [5], where the authors present real- time extension of UML statecharts, but their semantics is substantially different. From the statecharts syntax we borrow only graphical representation of the hierarchical states (sequential and parallel) and terminology (OR-state, AND-state).
The paper is organized as follows. In section 2, we review timed HSM and their syntax. In section 3, we give the operational semantics of THSM, and in section 4, the compositional semantics of traces of THSM. Finally in section 5, we discuss the computational complexity of the reachability problem for hierarchical time models and the task how to solve this problem.

## 2. FORMAL STRUCTURE OF TIMED HSM

Among many variants of definitions of finite-state machines, the authors of [1] choose a definition in which edges are labeled with alphabet symbols. A finite-state machine consists of a finite set $Q$ of states, a finite alphabet $E$ of symbols, a set $Q_0$ of initial states, a set $Q_F$ of final states, and set $T \subseteq Q \times E \times Q$ of transitions. For modeling hierarchy and concurrency, they proposed a machine, which states can be other machines arbitrarily nested, and which

can be composed from a set of component machines, which synchronize on transitions labeled with common alphabet symbols. In the following definition, we extend the hierarchical state machine (HSM) with time variables called clocks so that, we will be able to model real-time systems.

**Definition 1**
A Timed HSM is a structure THSM = ( $X$, $Q$, $Q_0$, $Q_F$, $E$, $\delta$, $\rho$, $inv$, $T$) where:
$X$ is a finite set of time variables with values from $R^+$ also called clocks,
$Q$ is a finite set of discrete states,
$Q_0 \subset Q$ is a finite set of the initial states,
$Q_F \subset Q$ is a finite set of the final states,
$E$ is a finite set of event symbols (alphabet),
$\rho$ is the hierarchy relation on $Q$,
$\delta$ is a default entrance function,
$inv$ is a function associating with each discrete state $q \in Q$ a convex $X$- polyhedron[1] called invariant of $q$,
$T \subseteq Q \times L \times Q$ is a finite set of transitions, where $L$ is a set of labels $l = (\zeta, e, X_0)$, where $\zeta$ is a conjunction of atomic constraints on $X$ defining a convex $X$- polyhedron, called the guard of transition, $e \in E$ is an event symbol, $X_0 \subseteq X$ is a set of clocks to be reset by taking the transition.

A. States
$Q$ is a finite set of discrete states consisting of the subset $Q^+$ of serial or OR- states, the subset $Q^\times$ of parallel or AND- states, and the subset $Q^{basic}$ of basic states. The hierarchical structure of THSM states is represented by the binary relation $\rho$ on $Q$ and satisfies the following conditions:

- There exists a unique state $r$, called the root state of Timed HSM, such that for no state $q \in Q$, $(q \rho r)$.
- For every state $q \in Q$, $q \neq r$, there exists a unique state $p \in Q$ such that $(p \rho q)$. The state $p$ is called an immediate super-state of $q$, whereas $q$ is an immediate sub-state of $p$.
- A state $q \notin Q$ has no immediate sub-states if and only if $q$ is basic $q \in Q^{basic}$.
- If $(p \rho q)$ then either $p \in Q^+ \wedge (q \in Q^+ \vee q \in Q^\times \vee q \in Q^{basic})$ or $p \in Q^\times \wedge (q \in Q^+ \vee q \in Q^{basic})$.

The default entrance function $\delta$ is mapping $\delta: Q^+ \rightarrow Q_0$ where for every $p \in Q^+$ and $q^I \in Q_0$ such that $(p \rho q^I)$, $q^I$ is the default initial sub-state of the super-state $p$.
A state of Timed HSM is a pair $(q, v)$, where $q \in Q$ is a discrete state and $v \in inv(q)$ is a $X$-valuation satisfying the invariant of $q$. $X$-valuation is a function $v: X \rightarrow R^+$ assigning to each clock $x \in X(q)$ a non-negative real value $v(x)$, where $X(q)$ denotes the clocks in the scope of the discrete state $q$. If discrete state $q$ has not assigned an invariant then the time can progress in $q$ without bounds.
If $q$ is OR- state with immediate sub-states $q_1,...,q_k$, then

$$inv(q) \supseteq \bigcup_{i=1}^{k} inv'(q_i) \qquad (1)$$

[1] $X$-polyhedron is the intersection of an atomic constraint on $X$, which are in the form $x \sim c$ or $x\text{-}y \sim c$, where $x$, $y \in X$, $\sim \in \{<, \leq, \geq, >\}$ and $c$ is a positive integer constant.

where $n$ is a number of such sub-states of $q$, for which there exists an input transition with label $l$, in which the set $X_0 \neq 0$ (at least one clock is reset), and

$$inv'(q_i) \supseteq \bigcup_{j=i}^{n_i} inv(q_j),$$

where $n_i$ is a number of sub-states between $q_i$ and $q_{i+1}$ for which $X_0 = 0$.
If there is a finite loop in the OR-state, the invariant of the states from the loop is multiplied by the number of cycles in the loop.
For example, if the constraints of clocks of each sub-state $q_i$ are $X$-hyperplanes[2] $x_i < c_i$, $i=1,...,k$ then the invariant of state $q$ will be $inv\ (q) \equiv x < c$, where

$$c \geq \sum_{i=1}^{n} c_i',$$

where $n$ in as in (1) and $c'_i = \max_{i \leq j \leq n_i} c_j$.
If $q$ is AND- state with immediate sub-states $q_1,...,q_k$, then

$$inv(q) \supseteq \bigcap_{i=1}^{k} inv(q_i) \qquad (2)$$

Similarly, if the constraints of clocks are as in the previous case, then for AND-state the invariant of state $q$ can be $inv(q) \equiv x < c$, where $c \geq \min_i(c_i)$.

B. Transitions
Consider a state $(q, v)$. Given a transition $t=(q, guard(t), e, X_0, q')$ such that $v \in guard(t)$ and $v' = v\ [x:=0, \forall x \in X_0] \in inv(q')$, $(q, v) \rightarrow^e (q', v')$ is a discrete transition of Timed HSM also called the $e$-successor of $(q, v)$. The condition $v \in guard(t)$ is also called an enabling condition of transition $t$. We assume that for each transition $t$ with the source discrete state $q$ $guard(t) \cap inv(q) \neq 0$ and for some outgoing transition $t$ of discrete state $q$ if $X$-valuation $v \notin inv(q) \Rightarrow v \in guard(t)$.
If $q$ or $q'$ are the hierarchical discrete states, the transition $t$ is also called a top-level transition of Timed HSM. It connects the final sub-state of $q$ with the initial sub-state of $q'$ given by a default entry function $\delta(q')$. The final sub-state of an OR-state is a state without output transition or if there is a loop within OR-state then a final sub-state is the final state of the loop (its output transition connects it with the initial state of the loop, e.g., states $c$, $m$, $w$ from Fig. 1). In the case, if $q$ is AND-state with the immediate sub-states, which can be OR-states or basic states, the final states are stated as before, and taking the transition means a joint exit from each of its orthogonal components. This top-level transition is synchronization transition since all orthogonal sub-states of an AND-state must be synchronized on their final states. This requirement is in contradiction with (2) because independent transitions of components may interleave, and interleaving implicitly allows indefinite waiting of a component before achieving synchronization. In spite of that, a synchronized exit from AND-state is an important assumption to introduce the compositional semantics of THSM, as will be shown in the next sections. The above contradiction with (2) can be turned by introducing waiting state as a final state to all "faster" orthogonal components (see sub-state j of a from Figure 2).
If $q'$ is AND- state, the initial states for each sub-state $q_i'$, $i=1,2,..,k$ of $q'$ are given by default entry function $\delta(q_i)$, and taking the transition means a fork entrance to each of its orthogonal components. If $q$ and $q'$ are both AND-states, taking the transition means a

[2] $X$-hyperplane is a set of valuations satisfying an atomic clock constraint.

joint exit from each orthogonal component of $q$, and fork entrance to each orthogonal component of $q'$. The transitions between top-level states have a lower priority compared to the inner-level state transitions, if they are enabled simultaneously. For example, the transition *t2* between states *u* and *m* from Fig.1 has lower priority compared to any transition of state *u*. This assumption gives the necessary condition for the synchronized exit from all orthogonal components of AND-state. The state consistency, which means that the control is always passed back to super-states, is ensured by no infinite loops within a sub-state. In contrast with statecharts we do not allow inter-level transitions, i.e., the transitions crossing the borderline of states.

A time transition from $(q, v)$ has the form $(q, v)\rightarrow^{\tau}(q, v+\tau)$ where $\tau\in R^{+}$, and $v+\tau\in inv(q)$, and it means that system is being in the state $q$ while time elapses. If $q$ is a hierarchical state the time $\tau$ is bounded according to the relations (1) or (2) in accordance with its type.

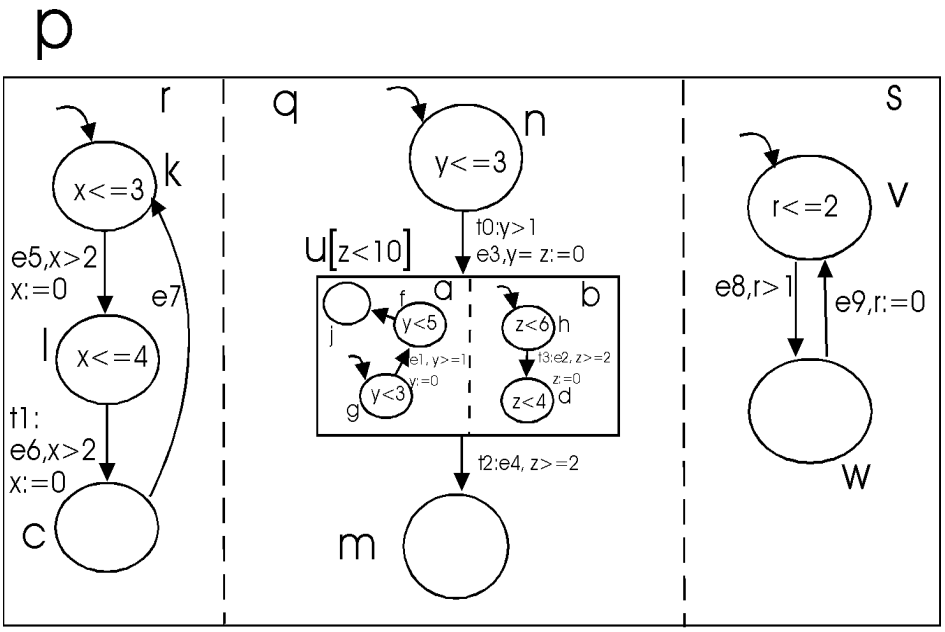


Fig.1.Timed HSM

Example.

As an illustration, see Figure 1 showing a partial THSM. The root state $p$ is product of three OR-states *r*, *q*, and *s*. The state $q$ is mapped to lower level sub-states *n*, *u*, and *m*. For instance, sub-state *u* is AND- state too. The hierarchical structure of THMS can be represented by directed acyclic graph (DAG) [2]. The terminal nodes correspond to the basic states; the internal nodes may correspond to the hierarchical states. Two important parameters of DAG are its width and depth; width of DAG is the maximum number of components in orthogonal nodes, and depth is the length of the longest path in the DAG. The DAG of THSM from Figure 1 is in Figure 2. An invariant of state is given either in the position of the state or beyond the name of state in the square brackets. For example, state *u* has invariant *X*-polyhedron given as intersection of constraints $t< 8$ and $z< 6$; state *l* has invariant *X*-hyperplane $x< 4$. The timed transition must satisfy the constraint given by invariant, e.g., the model can be maximally 4 time unit in *l*, so $(l, 0)\rightarrow^{4}(l, 4)$ is legal transition but $(l, 0)\rightarrow^{5}(l, 5)$ is not. The discrete transition between states is given by an oriented curve, and can be labelled with the event, the guard condition, and by the reset function. For example, the transition *t1* from Figure 1 is taken if both the guard $x> 2$ is satisfied (enabling condition), and event *e6* occurs; then the clock $x$ is set to zero. Default initial states of the hierarchical states are marked by small arrow, e.g., states *k*, *n*, *v*, *g*, and *h*. For example, when the transition *t0* is

taken, the model enters the default initial states *g*, and *h* of super-states *a* and *b* simultaneously. On the contrary, when transition *t2* is taken, the model must leave the final sub-states *j* and *d* of states *a*, and *b*.

## 3. OPERATIONAL SEMANTICS OF TIMED HSM

**Definition 2**

Let $s = < s_1,s_2,...,s_n>$ be a tuple of basic states $s_i = (q_i, v)$ where $q_i$ is a basic discrete state and $v$ is a valuation satisfying the invariant of $q_i$, $i= 1,...,n$ then s will be called a configuration if every pair of discrete states $q_i$, $q_j$ in $s$ is orthogonal, i.e., their lowest common ancestor[3] is an AND-state.

In Figure 1, the tuple $<k, n, v>$ is configuration but $<k, f, m, v>$ is not, since the pairs of basic states $f, m$ is not orthogonal.

The configuration of state $q$ is full if it is maximal, i.e., it contains maximal number of basic states that the system can be in simultaneously.

The set $\Re_s$ of full configurations for $s =(q, v)$, $q\in Q$ can be computed inductively as follows:

1) If $q$ is basic state then $\Re_s =\{<s>\}$
2) If $q$ with sub-states $q_1,...,q_k$ is OR- state then $\Re_s =\{<\Re_{si}>\}$
3) If $q$ with sub-states $q_1,...,q_k$ is AND- state then $\Re_s =\{<\Re_{s1}\times...\times\Re_{sk}>\}$

Example:

The full configurations of state $q$ from Figure 1 are for example: $<k, <f, h>, v>$, $<l, <g, h>, w>$, but the configuration $<k, n, v>$ is not full.

A. Synchronization

Let $s= <s_1,...,s_i,s_j,...,s_n>$ and $s'= <s_1...,s_i',s_j',...,s_n>$ be two configurations, and event $e_i=e_j=e\in l_i\wedge l_j$, where $l_i$, $l_j$ are labels of transitions $t_i$, $t_j$ ; then the synchronization of discrete transitions $t_i$, $t_j$ with scopes $\chi(t_i)\neg\rho\ \chi(t_j)$[4], where $t_i=(s_i, guard(t_i), e, X_i, s_i')$, $t_j= (s_j, guard(t_j), e, X_j, s_j')$ yields the transition $t_i \parallel t_j =(s, guard(t_i)\cap guard(t_j), e, X_i\cup X_j, s')$, where operator $\parallel$ means simultaneous execution of both transitions. Also event $e$ is called a synchronization event.

B. Interleaving

Let $s$ and $s'$ be two configurations as formerly. Let $e_i \in l_i \neq e_j \in l_j$, i.e., $t_i=(s_i, guard(t_i), e_i, X_i, s_i')$, $t_j= (s_j, guard(t_j), e_j, X_j, s_j')$, then the interleaving of $t_i$, $t_j$ yields the transitions $t_i \parallel \perp = (s, guard(t_i), e_i, X_i, s')$ with $s_j = s_j'$ for $j\neq i$, and $\perp \parallel t_j = (s, guard(t_j), e_j, X_j, s')$ with $s_i = s_i'$ for $i\neq j$, where $\perp$ is an empty symbol.

C. Timed transition

Let $s = <s_1,...,s_i,s_j,...,s_n>$ be a configuration, then time transition $s_i\rightarrow^{\tau} s_i+\tau$, for some $i\in \{1,...,n\}$ has the result that an amount of time $\tau$ passes in the configuration $s$ only if all components can delay $\tau$ time units, i.e., $s+\tau =<s_1+\tau,...,s_i+\tau...,s_n+\tau>$ and we simply write $s\rightarrow^{\tau} s+\tau$.

---

[3] The lowest common ancestor (lca) of states $q_1$ and $q_2$ is a state $q$ such that $q_1$, $q_2$ are sub-states of $q$, and for every sub-state $p$ of $q$ either $p$ is super-state of $q_1$ and $p$ is not super-state of $q_2$ or opposite.

[4] The scope of transition t is denoted $\chi(t)$ and represents serial lca of source and target discrete states of transition, and $\chi(t_i)\neg\rho\chi(t_j)$ means that serial lowest common ancestors of both transitions are not in the hierarchical relation $\rho$.

D. Paths and traces

Given a word $\sigma = e_0\, e_1 \ldots e_n$ over the alphabet $E$, an accepting path of Timed HSM over $\sigma$ is a finite sequence $\pi = s_0 \rightarrow^{\pi 0} s_0 + \tau_0 \rightarrow^{e0} s_1 \rightarrow^{\tau 1} s_1 + \tau_1 \rightarrow^{e1} s_2 \rightarrow^{\tau 2} \ldots \rightarrow^{en} s_{n+1}$, such that $s_0 = (q_0, v)$, $q_0 \in Q_0$ is an initial configuration, $s_{n+1} = (q_{n+1}, v)$, $q_{n+1} \in Q_F$ is a final configuration, and for all $i = 0,1,2, \ldots, n$, $s_i + \tau_i$ is the $\tau_i$- successor of $s_i$, and $s_{i+1}$ is $e_i$- successor of $s_i + \tau_i$.

A word $\sigma$ is also called a trace of THSM. The set of all traces of THSM, which have an accepting path, is called the language of THSM, denoted $\Sigma$.

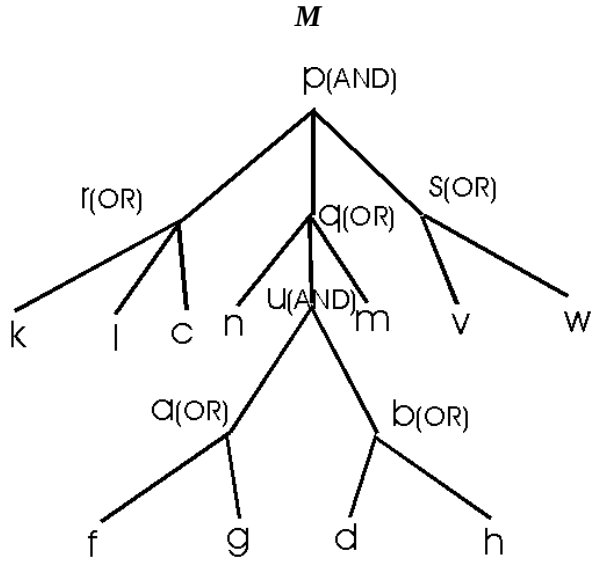


Fig. 2. DAG of THSM from figure 1

## 4. COMPOSITIONAL SEMANTICS OF TRACES

The goal is to define such compositional semantics of traces, which ensures that the semantics of the whole system can be determined from the semantics of its components. In order to show that our semantics is compositional, we need to be able to define the semantics of a hierarchical state only in terms of trace semantics of its sub-states.

We show how traces can be defined with further hierarchical extension of state $q$.

**Lemma 1 (Trace construction)**

Let $\sigma \in \Sigma_q$ be a trace of state $q$. Let $p$ be OR-state extension of $q$ ($p \rho q$); then $\sigma' \in \Sigma_p$ looks like $\sigma' = \sigma_1\, \sigma_2 \ldots\, \sigma_i\, \sigma_{i+1} \ldots \sigma_k$, where $\sigma_j$, $j = 1, 2, \ldots, k$, $j \neq i$ are traces of new components $p_j$ of state $p$, and $\sigma_i = \sigma$ is an original trace of state $q$.

The lemma says that $\sigma'$ arises from $\sigma$ by the concatenation of traces of the individual sub-states of OR- state $p$ for every accepting path of each sub-state. In this case, top-level transitions between sub-states connecting the entrance configuration of successor state with the exit configuration of the predecessor state can be executed immediately (they are not labelled with events and guards).

Now, consider the case when $p$ is AND-state extension of $q$ arising by adding $k$ orthogonal components $p_i$ to state $q$ then the accepting path $\pi' = s_0' \rightarrow^{\pi 0} s_0' + \tau_0 \rightarrow^{e0} s_1' \rightarrow^{\tau 1} s_1' + \tau_1 \rightarrow^{e1} s_2' \rightarrow \ldots \rightarrow^{em-1} s_m'$, where $s_i'$ are the new configurations of state $p$. Now, the new trace $\sigma'$ cannot be expressed directly as the concatenation of the original trace $\sigma$ of $q$ *with* traces of other sub-

states. The reason is that first, the full configurations of states $p$ and $q$ are different and second, some new transitions arising with the state extension can be in synchronization with transitions of state $q$.

For the special case of THSM called asynchronous THSMs, we will show the compositional semantics of traces also for AND- state extension of $q$. Asynchronous THSMs are characterized by sparse interaction between parallel components. That is, there is no synchronization of orthogonal components by means of shared events; all interactions are assumed to be modelled either by high-level transitions or by the transition constraints. We will use the following lemmas.

**Lemma 2**
With AND- state extension of state $q$ its configuration is extended with a tuple of basic states of the new orthogonal components.
**Proof**
It is evident from the construction of full configuration.

**Lemma 3**
Let $p$ be AND-state extension of state $q$ ($p \rho\, q$). If $E_{p\text{-}q} \cap E_q = 0$ (the set of common events of added orthogonal state components and state $q$ is empty) then $\sigma'$ is concatenation of $\sigma$ with the trace formed by interleaving of transitions $t_i$, $t_j$ with scopes $\chi(t_i)=p\text{-}q$ and $\chi(t_j)= q$ with the difference that now the traces $\sigma'$ and $\sigma$ are constructed from the accepting path containing the extended state configurations according to Lemma 2.
**Proof**
We mark the configuration of $p$ as $s_p=<<s_1>,<s_2>,...,<s_q>>$, where $<s_1>,<s_2>,...$ are configurations of new orthogonal components of state $p$, and $<s_q>$ is the configuration of state $q$. We want to show that the accepting path $\pi' = s_{p0} \rightarrow^{\pi p0} s_{p0} + \tau_{p0} \rightarrow^{ep0} s_{p1} \rightarrow^{\pi p1} s_{p1} + \tau_{p1} \rightarrow^{ep1} \rightarrow s_{p2} \rightarrow^{\pi p2} .... \rightarrow^{epm\text{-}1} s_{pm}$ of state $p$ contains the original accepting path of state $q$, $\pi = s_{q0} \rightarrow^{\pi q0} s_{q0} + \tau_{q0} \rightarrow^{eq0} s_{q1} \rightarrow^{\pi q1} s_{q1} + \tau_{q1} \rightarrow^{eq1} s_{q2} \rightarrow^{\pi q2} ... \rightarrow^{eqn\text{-}1} s_{qn}$, where $m \geq n$.

- The time transition of $\pi$ $s_{qi} \rightarrow^{\pi qi} s_{qi} + \tau_{qi}$ is contained in $s_{pj} \rightarrow^{epj} s_{pj+1} \rightarrow^{\pi pj+1} s_{pj+1} + \tau_{pj+1}$ because $s_{qi} \subset s_{pj} \wedge s_{qi} \subset s_{pj+1}$ and $s_{qi} + \tau_{qi} \subset s_{pj+1} + \tau_{pj+1}$, where event $e_{pj} \notin E_p$.
- The discrete transition of $\pi$ $s_{qi+1} \rightarrow^{eqi} s_{qi+2}$ is contained in $s_{pj+1} \rightarrow^{\pi pj+1} s_{pj+1} + \tau_{pj+1} \rightarrow^{epj+1} s_{pj+2}$ because $s_{qi+1} \subset s_{pj+1} \wedge s_{qi+1} \subset s_{pj+1} + \tau_{pj+1}$ and $s_{qi+2} \subset s_{pj+2}$, where $e_{pj+1} = e_{qi}$.

According to the assumption $E_{p\text{-}q} \cap E_q = 0$ all discrete state transitions of state $p$ are interleaving, it means that all events $e \in E$ causing the discrete transitions in the path $\pi'$, do not influence the components $<s>$ of configurations from $\pi'$ which are changing in accordance with the configurations from $\pi$. That means now, the trace $\sigma'$ will contain the original trace $\sigma$. In the case when $E_{p\text{-}q} \cap E_q \neq 0$ ( the synchronization of orthogonal components by means of shared events), the enabling conditions of transition from the original path $\pi$ will be different (the conjunction of guards of transitions with shared event, see the synchronization case above) than in the case without state extension. That means now, the trace $\sigma$' will not contain the original trace $\sigma$.

The above lemma is used to prove the following theorem:

**Theorem 1**
The set of traces $\Sigma$ of THSM can be computed from the set of traces of its sub-states and its discrete and timed transitions.
Proof.
It follows immediately from the preceding lemmas.

### 4.1 Refinement

The trace semantics allows us to define refinement between THSM. The refinement relation between models captures the notion that two THSMs describe the same system at different levels of detail.

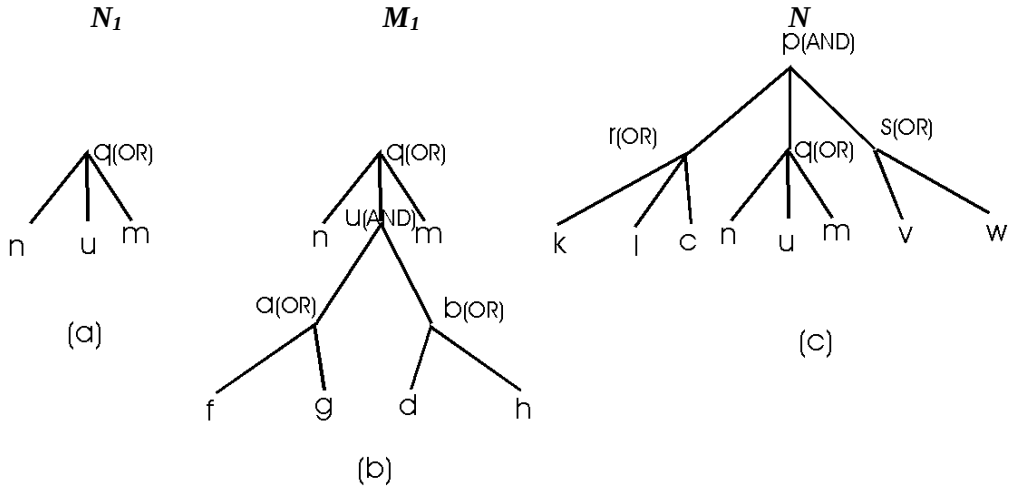


Fig. 3. DAGs showing the refinement of the hierarchical states

**Definition 3**

Let $M_1$= ( $X$, $Q_1$, $Q_0^1$, $Q_F^1$, $E$, $\delta$, $\rho$, $inv_1$, $T_1$) and $M2$= ( $X$, $Q_2$, $Q_0^2$, $Q_F^2$, $E$, $\delta$, $\rho$, $inv_2$, $T_2$) be two THSM such that $Q_1 \subseteq Q_2$ , $Q_0^1 \subseteq Q_0^2$, $Q_F^1 \subseteq Q_F^2$, $T1 \subseteq T_2$, and for all $q \in Q_1 \cap Q_2$, $inv_1(q) = inv_2(q)$ then $M_2$ refines $M_1$, denoted $M_2 \preceq M_1$, if $\Sigma_2 \subseteq \Sigma_1$.

The sub-machine $M' = (X, Q', Q_0', Q_F', E, \delta, \rho, inv', T')$ of THSM $M = (X, Q, Q_0, Q_F, E, \delta, \rho, inv, T)$ can be defined by the following condition: $\exists\, q\ p,\ q \in Q \wedge q \notin Q \cap Q' \wedge p \in Q \cap Q'$ such that $q$ is a immediate super-state of $p$ ($q \rho\, p$). We mark by $C_M[M']$ the context of $M'$ in $M$.

We show that refinement operator is compositional with respect to the hierarchy relation of THSM.

**Theorem 2**

Given a THSM $M$ with sub-machine $M_1$, such that

(a1) $M_1 \preceq N_1$ , where $N_1$ is a sub-machine of $N$ and

(a2) the $C_M[M_1] = C_N[N_1]$ then

$M \preceq N$.

Proof.

It follows from:

$M \preceq C_M[M_1] \preceq C_M[N_1] \preceq C_N[N_1] \preceq N$, where the second refinement relation results from theorem assumption (a1) and the third relation results from the theorem assumption (a2).

Example:

As an illustration of the theorem we introduce an example of DAG from Figure 2. DAG of $N_1$ with state $u$ that is not refined, is given in Figure 3(a). DAG of $M_1$, now with state $u$ refined to its sub-states is given in Figure 3(b). $N$ is given in Figure 3(c). You can see that DAG of $M$ from Figure 2 refines $N$ that is the consequence of the refinement of $N_1$ with $M_1$.

## 5. REACHABILITY

The reachability problem belongs to the problems, which are widely discussed in many works concerning the formal methods in the model verification. The safety properties of the model can be also transformed to the reachability problem, which can be then formulated as: Is a state, in which the safety properties are not satisfied, reachable or not. The reachability problem can be formally defined as:

Given an initial configuration $s_0$ and the set $P$ of final configurations of THSM. Our objective is to verify whether there exists a configuration $p \in P$ and a path $\pi = s_0 \rightarrow s_1 \rightarrow,\ldots,\rightarrow p$ such that $p$ is $\pi$-reachable from $s_0$.

Authors of [1] discussed the reachability problem of communicating HSM (CHSM) from the point of view of its complexity. In CHSM, the concurrency and hierarchy operator are arbitrary nested, and the product components can synchronize with each other at different levels of hierarchy. These features make the reachability problem significantly difficult to solve. Authors show that reachability problem for a CHSM can be solved in time $O(n^{m^d})$ where $n$ is number of states, $m$ is a width and $d$ is a depth of DAG. The complexity is EXPSPACE- complete. The authors define well-structured CHSM the complexity of which is lower. The restriction to well-structured machines ensures that if two or more hierarchical machines are composed together, then they can synchronize only at the top level. The reachability problem in this case is PSPACE-complete and can be solved in time $O(k.n^m)$ where $k$ is number of hierarchical machines, and $n$ maximal number of states of the hierarchical machine.

Time extension of the HSM is not elaborated to this time, but there is a very nice developed theory of timed automata in the literature. Timed automaton is specification formalism without the state hierarchy and parallelism. Its state structure is very similar to our OR-state with the sub-states, which are basic. By introducing time into the model with the hierarchical states, makes the verification problem more difficult. The reason is that, now, the state space is not only hierarchically structured but it becomes also infinite. There are several techniques solving the approximation of infinite state space by its finite representation. For timed automata the known finite partition of the state space are symbolic states for example regions or zones. The symbolic state is an abstract state containing a discrete state together with a time region or zone where all time-abstracting bisimulation equivalent valuations of clocks belong to. The precise definition is in [6]. In [4], the computational complexity of the parallel composition of $k$ timed automata is given. The reachability problem is PSPACE- complete and can be solved in time $n^{k+1}.2^{O(k.l.\log(kcr))}$, where $l$ is number of clocks, and every constant in clock constraints is bounded by $c$.

Our compositional semantics of THSM allows us to decompose the complex model to the sub-models, which have lower state space in accordance with the refinement operator, which is compositional with respect to the hierarchy relation (including the operator of the state parallel composition). The complex reachability problem can be such divided into sub-problems the computational complexity of which is lower.

For example, the reachability problem of the example from Figure 1 is to show that the final configuration <c, m, w> can be reached from the initial configuration <k, n, v>. The problem can be divided into sub-problems: to show the reachability of the final configuration for each parallel component of state p individually. Of coarse, due to timing constraints (invariants in states and guards in transitions) the final configurations for each parallel component need not be reached at the same time. This problem can be solved by some technique known from the theory of timed automata [6], e.g., by finite approximation of the infinite configuration space with zones.

## 6. CONCLUSION

We show how the compositional semantics of timed HSM can be exploited in the reachability analysis without translation of the model to its flattened version. In the future our work, we plan to concentrate on the wide class of verification tasks concerning mainly the verification of the temporal logic formulas expressing the real-time model behaviours.

*The author is grateful to the Slovak Grant Agency for Science VEGA (grant No, 2/1101/22) for partial supporting of this work.*

## REFERENCES

[1] Alur R., S. Kannan, and M. Yannakis: Communicating Hierarchical State Machines. Proceedings of the 26th International Colloquium on Automata, Languages, and Programming, LNCS 1644, Springer- Verlag, 1999, 169--178.
[2] Brave Y. and M. Heymann: Control of Discrete Event Systems Modeled as Hierarchical State Machines. IEEE Transaction on Automatic Control, Vol. 38, No. 12, 1993, 1803-1819.
[3] Alur R., D. Dill: A theory of timed automata. Theoretical Computer Science. 126, 1994, 183- 235.
[4] Alur R., Timed Automata. 11th Conference on Computer- Aided Verification, LNCS 1633, Springer-Verlag, 1999, 8-22.
[5] David A., M. O. Moller, and Wang Yi: Formal Verification of UML Statecharts with Real-Time Extensions. Proceedings of FASE 2002, LNCS 2306, 2002, 218-232.
[6] Tripakis S., S. Yovine: Analysis of Timed Systems using Time- Abstracting Bisimulations. Formal Methods in System Design, 18, Kluver Academic Publishers, 2001, 25- 68.