Timed Automata with Data Structures for Distributed Systems Design and Analysis

Ruggero Lanotte
Dipartimento di Scienze della Cultura, Politiche e dell'Informazione
Università dell'Insubria

Andrea Maggiolo-Schettini and Angelo Troina
Dipartimento di Informatica - Università di Pisa

Abstract

Systems of Data Management Timed Automata (SDMTAs) are networks of communicating timed automata with structures to store messages and functions to manipulate them. We prove the decidability of reachability. As an application, we model and analyze a cryptographic protocol.

1 Introduction

Automata based formalisms have been successfully used to describe concurrent and distributed systems and analyze their properties. Basic formalisms allow describing the nondeterministic behavior of a system. This is enough to model the functionality of systems, and hence to be used to capture also the logical information flow, but if one wants to capture also time-dependent information flow, formalisms must be extended with real time features. Alur and Dill have proposed Timed Automata [2]. In general, timed automata models have an infinite state space. However the region automata construction shows that this infinite state space can be mapped to an automaton with a finite number of equivalence classes (regions) as states, and finite-state model checking techniques can be applied to the reduced, finite region automata. Among the model checkers for timed automata we quote Kronos ([11]) and UPPAAL ([3]). Systems of communicating agents can be described by automata composed in parallel and sharing synchronization channels. Transitions labelled with a complementing channel name can be taken at the same moment and data transmission is typically modelled by a synchronization, where global variables are updated ([5]). In [9] the authors propose a model of communicating automata tailored for describing and analyzing protocols with timing constraints and verifying their security. Each participant in the protocol is described by a state transition diagram where transitions are labelled by events which represent the sending of a structured message along a channel. The performance of a transition is conditioned by the trigger of a communication event and by the temporal constraints imposed by a delay and/or timeout. Communication is synchronous. Primitive messages (public/private keys, identities, nonces, etc.) can be composed by using cryptographic primitives (encryption, hashing, signature, etc.). An initial evidence function assigns each initial state the set of evidence that is known by each participant at the beginning. Such evidence can be augmented by receiving messages from other participants and is reset to the initial conditions every time an input state is reached. The semantics of these descriptions is given in terms of Timed Automata, thus obtaining a specification on which properties can be verified.

In this paper we define Systems of Data Management Timed Automata (SDMTAs). An SDMTA has a finite set of elementary messages, a finite set of functions to elaborate them and a finite set of Data Management Timed Automata (DMTAs). Each DMTA has a finite set of channel labels, a finite set of clocks, a finite set of states (one of them is the initial state), an initial condition and an initial knowledge, a finite set of transitions. Transitions from state to state of a DMTA represent either an internal move with the computation of a term (which enriches the knowledge of the automaton) or the input or the output of a term on a channel. The performance of a transition is conditioned by the fulfillment of a constraint. Two DMTAs of a system may perform a communication step modelling the communication of a term through a channel. Time elapsing is modelled by a timed step of the automata of the system.

Our formalism extends the formalism of [9] to deal with a general class of distributed communicating systems with data structures. Note that the formalism of [9] assumes a bounded memory and has the power of regular languages, while our formalism assumes an unbound memory and,

when endowed with a concept of recognized language, does accept languages which are not regular. Differently with respect to [9], we define a direct operational semantics of SDMTAs in terms of steps and runs, and we prove the decidability of the reachability. This allows proving properties expressible in terms of reached states. As an application, we show how we can model cryptographic protocols and formalize and decide a secrecy property. We do it for a version of the known Yahalom protocol adapted to take timeouts and retransmissions into account.

We leave as a future work to develop a concept of bisimulation for SDMTAs, and to study expressiveness power of our model. We believe that a model which maintains simplicity and visual effectiveness of automata but does not abstract from data structures and still allows deciding interesting properties, has many applications besides the kind of the one shown in this paper as an example.

2 Basic notions

Let us assume a set of X positive real variables x called *clocks*. A *valuation* over a set of clocks is a mapping $v : X \to \mathbb{R}^{\geq 0}$ assigning real values to clocks. For a valuation v and a time value $t \in \mathbb{R}^{\geq 0}$, let $v + t$ denote the valuation such that $(v + t)(x) = v(x) + t$, for each clock $x \in X$.

Let $C = \{C_1, \ldots, C_m\}$ be a finite set, where C_i denotes a finite set of elementary messages. For an elementary message we mean non composed/manipulated message (i.e., names are elementary messages, lists of elementary messages are not).

Let us assume a set Υ of message variables μ that can assume values in $(\cup_{j=1}^{n} C_j) \cup \mathbb{N}$. Given a finite set of message variables, an instance I relates a message variable μ to a value in $(\cup_{j=1}^{n} C_j) \cup \mathbb{N}$. Namely, $I : \Upsilon \to (\cup_{j=1}^{n} C_j) \cup \mathbb{N}$.

Let $\Omega = \{f_1, \ldots, f_n\}$ denote a finite set of functions. Given a finite set of message variables Υ, the set of terms $\mathcal{T}(\Upsilon)$ is defined as:

$$\tau ::= c \mid w \mid \mu \mid f(\tau_1, \ldots, \tau_k)$$

where $c \in C_i$ for some i, $w \in \mathbb{N}$, $\mu \in \Upsilon$ is a message variable, f is a function in Ω with arity k.

We use C and Ω to represent a set of data structures and a set of functions to manipulate such structures. In general, C may be any set of data structures of different types and Ω may be any set of functions which represent operations on such structures. The examples we shall use to explain the framework focus on the application of our model to the case of cryptographic security protocols. In such a context, the set C may contain sets of ground messages exchanged within the protocol (for example a set of plaintext messages, a set of agent names, a set of keys...), and the set Ω may contain the basic cryptographic primitives, as message pairing, encryption, nonce generation, hashing, etc.

Example 2.1 *Consider $C = \{A, M, K\}$, where $A = \{a, b, \ldots\}$ is a set of agent names, $M = \{m_1, m_2, \ldots\}$ is a set of basic messages (i.e. a set of plaintext messages represented by bitstrings of a fixed length) and $K = \{k_1, k_2, \ldots\}$ is a set of keys. Given a finite set of message variables Υ, we assume $\Omega = \{Pair, Enc, Nonce\}$, with domains $\mathcal{T}(\Upsilon) \times \mathcal{T}(\Upsilon)$, $\mathcal{T}(\Upsilon) \times K$ and $(A \cup \Upsilon) \times (\Upsilon \cup \mathbb{N})$, respectively. $Pair(\tau_1, \tau_2)$ denotes the concatenation of the terms τ_1 and τ_2, $Enc(m_1, k_1)$ denotes the encryption of message m_1 with the key k_1, and $Nonce(a, 100)$ denotes a nonce of the agent a with value 100. $Nonce(\mu_1, \mu_2)$, where $\mu_1, \mu_2 \in \Upsilon$ are variables, may be instantiated by $Nonce(a, w)$ for some $a \in A$ and $w \in \mathbb{N}$. $Enc(\mu, k)$, where μ is a variable can be instantiated by $Enc(m, k)$ for some $m \in M$, by $Enc(a, k)$ for some $a \in A$, by $Enc(k, k)$ for some $k \in K$ or by $Enc(w, k)$ for some $w \in \mathbb{N}$. We remark that μ_1, μ_2, μ cannot be complex terms, i.e. $Enc(Nonce(a, 1), k)$ or $Enc(Enc(m', k'), k)$ are not instances of $Enc(\mu, k)$.*

With $Var(\tau)$ we denote the message variables appearing in the term τ. For example $Var(Enc(Nonce(\mu_1, 100), \mu_2)) = \{\mu_1, \mu_2\}$.

We say that two terms τ and τ' in $\mathcal{T}(\Upsilon)$ are *reducible* (written $\tau \simeq \tau'$) if they have the same structure, namely $\tau \simeq \tau'$ if there exist $\mu_1, \ldots, \mu_n, \overline{\mu}_1, \ldots, \overline{\mu}_n \in \Upsilon$ such that $\tau = \tau'[\overline{\mu}_1/\mu_1] \ldots [\overline{\mu}_n/\mu_n]$.

Finally, with $\mathcal{K}$ we denote a knowledge. A knowledge $\mathcal{K}$ is a finite set of terms τ such that $Var(\tau) = \emptyset$.

Given a finite set of clocks X and a finite set of message variables Υ, we define the set $\Phi(X, \Upsilon)$ of *formulae* as follows:

$$\begin{aligned} \phi ::= \; & true \mid \tau \in \mathcal{K} \mid \tau = \tau' \mid \mu \in \mathbb{N} \mid \\ & x \sim c \mid x \sim y \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \end{aligned}$$

where $\phi, \phi_1, \phi_2 \in \Phi(X, \Upsilon)$, $\tau, \tau' \in \mathcal{T}(\Upsilon)$, $\mu \in \Upsilon$, $x, y \in X$, $\sim \in \{<, \leq, =, \geq, >\}$ and $c \in \mathbb{Q}$.

We will write $\tau \neq \tau'$ for $\neg(\tau = \tau')$, $\tau \notin \mathcal{K}$ for $\neg(\tau \in \mathcal{K})$, $\mu \notin \mathbb{N}$ for $\neg(\mu \in \mathbb{N})$, and $\tau \in \{\tau_1, \ldots, \tau_k\}$ for $\tau = \tau_1 \vee \cdots \vee \tau = \tau_k$. As an example, with $\tau \in C_i$, for some i, we denote the formula $\bigvee_{m \in C_i} \tau = m$.

Given a term τ and an instance I, we define the instantiation of τ as $I(\tau) = \tau'$ where τ' is the term resulting after replacing each μ syntactically occurring in τ with $I(\mu)$.

Let $\phi \in \Phi(X, \Upsilon)$, I be an instance, v a valuation of clocks and $\overline{\mathcal{K}}$ a knowledge; we say that I, v and $\overline{\mathcal{K}}$ *satisfy* ϕ,

written $I, v, \overline{\mathcal{K}} \models \phi$, in the following cases:

$$
\begin{array}{lll}
I, v, \overline{\mathcal{K}} \models true & & \\
I, v, \overline{\mathcal{K}} \models \tau \in \mathcal{K} & \text{iff} & I(\tau) \in \overline{\mathcal{K}} \\
I, v, \overline{\mathcal{K}} \models \tau = \tau' & \text{iff} & I(\tau) = I(\tau') \\
I, v, \overline{\mathcal{K}} \models \mu \in \mathbb{N} & \text{iff} & I(\mu) \in \mathbb{N} \\
I, v, \overline{\mathcal{K}} \models x \sim c & \text{iff} & v(x) \sim c \\
I, v, \overline{\mathcal{K}} \models x \sim y & \text{iff} & v(x) \sim v(y) \\
I, v, \overline{\mathcal{K}} \models \neg\phi_1 & \text{iff} & I, v \not\models \phi_1 \\
I, v, \overline{\mathcal{K}} \models \phi_1 \vee \phi_2 & \text{iff} & \text{either } I, v \models \phi_1 \text{ or } I, v \models \phi_2 \\
I, v, \overline{\mathcal{K}} \models \phi_1 \wedge \phi_2 & \text{iff} & \text{both } I, v \models \phi_1 \text{ and } I, v \models \phi_2.
\end{array}
$$

Example 2.2 *The formula $\mu_1 \neq \mu_2 \wedge \mu_1 \in \mathcal{K} \wedge \mu_2 \in \mathcal{K}$ says that in the knowledge $\mathcal{K}$ there are at least two elementary messages. Formula $\mu \in K$, where $K \in C$ denotes the set of keys, means that in $\mathcal{K}$ there is at least one key.*

3 Data Management Timed Automata

Given a finite set of clocks X (resp. message variables Υ) with X' (resp. Υ') we denote a new set of clocks (resp. message variables) such that $x' \in X'$ (resp. $\mu' \in \Upsilon'$) iff $x \in X$ (resp. $\mu \in \Upsilon$).

A System of Data Management Timed Automata (SDMTA) is a tuple $\mathcal{A} = (C, \Omega, A_1, \ldots, A_m)$, where:

- $C = \{C_1, \ldots, C_k\}$ is a set of elementary messages;
- $\Omega = \{f_1, \ldots, f_n\}$ is a set of functions;
- $A_1, \ldots, A_m$ are Data Management Timed Automata (DMTAs).

A DMTA is a tuple $A = (\Sigma, X, \Upsilon, Q, q_0, \phi_0, \mathcal{K}_0, \delta)$, where:

- Σ is a finite set of channel labels;
- X is a finite set of clocks;
- Υ is a finite set of message variables;
- Q is a finite set of states with $q_0 \in Q$ initial state;
- $\phi_0 \in \Phi(X, \Upsilon)$ is the initial condition;
- $\mathcal{K}_0$ is the initial knowledge of A.
- δ is a finite set of transitions. Each transition is a tuple (q, α, ϕ, q'), where $q, q' \in Q$ are the source and the target states respectively, for $a \in \Sigma$, $\alpha \in \{\epsilon(\tau), a?(\tau), a!(\tau)\}$ represents, respectively, the internal move producing the term τ, which enriches the knowledge of $\mathcal{A}$, or the input or the output of the term $\tau \in \mathcal{T}(\Upsilon)$ on channel a, ϕ is a formula in $\Phi(X \cup X', \Upsilon \cup \Upsilon')$. Variables in $X \cup \Upsilon$ and in $X' \cup \Upsilon'$ represent the value of variables before and after the firing of the transition, respectively.

An example of transition is $(q_0, a!(\mu), x < 5 \wedge \mu \in \mathcal{K} \wedge x' = x \wedge \mu' \in \mathbb{N}, q_1)$, stating that from the initial state q_0 the DMTA may reach state q_1 and output a message μ when this is in $\mathcal{K}$ and $x < 5$. The condition $x' = x$ means that the transition does not change the clock x, and the condition $\mu' \in \mathbb{N}$ means that the variable μ non-deterministically assumes a natural value after the transition.

3.1 Semantics

Given two valuations v_1, v_2 over X, with $v \oplus v'$ we denote the valuation on $X \cup X'$ such that for any $x \in X$, $(v_1 \oplus v_2)(x) = v_1(x)$ and $(v_1 \oplus v_2)(x') = v_2(x)$. Given two instances I_1, I_2 over Υ, with $I_1 \oplus I_2$ we denote the instance over $\Upsilon \cup \Upsilon'$ such that for any $\mu \in \Upsilon$, $(I_1 \oplus I_2)(\mu) = I_1(\mu)$ and $(I_1 \oplus I_2)(\mu') = I_2(\mu)$.

A configuration of an SDMTA $\mathcal{A} = (C, \Omega, A_1, \ldots, A_m)$ where $A_i = (\Sigma^i, X^i, \Upsilon^i, Q^i, q_0^i, \phi_0^i, \mathcal{K}_0^i, \delta^i)$, is a tuple $(s_1, \ldots, s_m)$ such that $s_i = (q, v, I, \mathcal{K})$ is a configuration of the DMTA A_i with $q \in Q^i$ a state of A_i, v a valuation over X^i, I an instance over Υ and $\mathcal{K}$ a knowledge.

Given two configurations $s = (s_1, \ldots, s_m)$ and $s' = (s'_1, \ldots, s'_m)$ such that $s_i = (q_i, v_i, I_i, \mathcal{K}_i)$ and $s'_i = (q'_i, v'_i, I'_i, \mathcal{K}'_i)$, we have that:

- there is a *ϵ-transition step* from s to s' (denoted $s \rightarrow_\tau s'$) if there exist an index i and $e = (q_i, \epsilon(\tau_1), \phi, q'_i) \in \delta^i$ such that $I_i(\tau_1) = \tau$, $(I_i \oplus I'_i), (v_i \oplus v'_i), \mathcal{K}_i \models \phi$, $\mathcal{K}'_i = \mathcal{K}_i \cup \{\tau\}$ and, for all $j \neq i$, $s'_j = s_j$;
- there is a *communication transition step* from s to s' with the term τ (denoted $s \rightarrow_{a(\tau)} s'$) if there exist two different indexes i and j, $(q_i, a!(\tau_1), \phi_1, q'_i) \in \delta^i$ and $(q_j, a?(\tau_2), \phi_2, q'_j) \in \delta^j$ such that:
  - $I_i(\tau_1) = I_j(\tau_2) = \tau$;
  - $(I_i \oplus I'_i), (v_i \oplus v'_i), \mathcal{K}_i \models \phi_1$ and $(I_j \oplus I'_j), (v_j \oplus v'_j), \mathcal{K}_j \models \phi_2$,
  - $\mathcal{K}'_i = \mathcal{K}_i \cup \{\tau\}$ and $\mathcal{K}'_j = \mathcal{K}_j \cup \{\tau\}$;
  - for all $k \notin \{i, j\}$, it holds that $s'_k = s_k$;
- there is a *timed step* from s to s' through time $t \in \mathbb{R}^{>0}$, written $s \rightarrow_t s'$, if, for any i, $q'_i = q_i$, $v'_i = (v_i + t)$, $I'_i = I_i$, and $\mathcal{K}'_i = \mathcal{K}_i$.

With the ϵ-transition step we model the internal data manipulation executed by the DMTA without any communication. For example, given a configuration $(q, v, I, \mathcal{K})$, the transition $(q, \epsilon(\mu), \phi, q')$, where $\phi = Enc(\mu, k) \in \mathcal{K} \wedge k \in \mathcal{K}$, states that if $(I \oplus I'), (v \oplus v'), \mathcal{K} \models \phi$ the DMTA may decipher a ciphertext $Enc(\mu, k)$ contained in its knowledge if also the key k is known, and then enrich the knowledge with the instantiation $I(\mu)$.

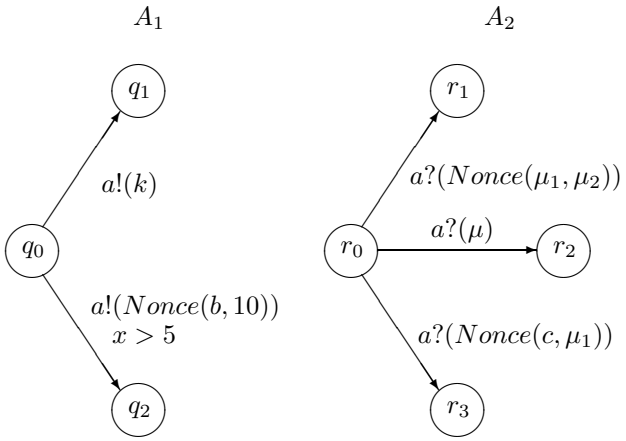

Figure 1. A System of Cryptographic Timed Automata.

A communication transition step $\rightarrow_{a(\tau)}$ models the communication of the term τ through the channel a. There is a synchronization between two DMTAs, and they both change their configuration by following the formula in the transition and by augmenting their knowledge. If variables appear in the output and input terms (τ_1 and τ_2, respectively), they should be *reducible* for transmission. In particular, we require that $I_i(\tau_1) = I_j(\tau_2)$ (also see Example 3.1). The other DMTAs of the system non involved in the communication remain in their original configuration.

Finally, a timed step models time elapsing. When time elapses, we reasonably assume that each DMTA A_i in the system performs a timed step by changing its valuation v_i.

Example 3.1 *Consider the DMTAs A_1 and A_2 in Figure 1, where q_0 and r_0 are the initial states of A_1 and A_2, respectively. If $\mathcal{K}_0$ is the initial knowledge of A_1 and we assume that $k \in \mathcal{K}_0$, the only communications that may happen between A_1 and A_2 through synchronization steps of matching terms are either $a(Nonce(b,10))$ (when x has value greater than 5) or $a(k)$. In both cases, the knowledge of A_2 is augmented with the term received by A_1.*

Given a DMTA $A = (\Sigma, X, \Upsilon, Q, q_0, \phi_0, \mathcal{K}_0, \delta)$, a configuration $s = (q_0, v, I, \mathcal{K}_0)$ of A is *initial* if $I, v, \mathcal{K}_0 \models \phi_0$. Given an SDMTA $\mathcal{A} = (C, \Omega, A_1, \ldots, A_m)$, we say that the configuration $s = (s_1, \ldots, s_m)$ of $\mathcal{A}$ is initial iff s_i is an initial configuration of the DMTA A_i for all $i \in [1, m]$.

A *run* of an SDMTA $\mathcal{A}$ is a finite sequence of steps $\sigma = s_0 \rightarrow_{\alpha_1} s_1 \rightarrow_{\alpha_2} \ldots \rightarrow_{\alpha_l} s_l$ where s_0 is an initial configuration of $\mathcal{A}$, s_j is a configuration of $\mathcal{A}$ and $\alpha_j \in \{\tau, a(\tau)\} \cup \mathbb{R}^{>0}$ for each $j \in [1, l]$.

A state q of of an SDMTA $\mathcal{A}$ is reachable iff there is a run $\sigma = (s_0^1, \ldots, s_0^m) \rightarrow_{\alpha_1} \ldots \rightarrow_{\alpha_l} (s_l^1, \ldots, s_l^m)$ of $\mathcal{A}$ such that there exist i and j with $s_j^i = (q, v, I, \mathcal{K})$.

3.2 Decidability of Reachability

We recall the definitions of clock equivalence [2]. Clock equivalence is a finite index equivalence relation permitting to group sets of evaluations and to have decidability results.

Let $\mathcal{A}$ be a DMTA; with $C_{\mathcal{A}}$ we denote the greatest constant that appears in $\mathcal{A}$.

Let us consider the equivalence relation $\approx$ over clock valuations containing precisely the pairs (v, v') such that:

- for each clock x, either $\lfloor v(x) \rfloor = \lfloor v'(x) \rfloor$, or both $v(x)$ and $v'(x)$ are greater than $C_{\mathcal{A}}$, with $C_{\mathcal{A}}$ the largest integer appearing in clock constraints over x;
- for each pair of clocks x and y with $v(x) \leq C_{\mathcal{A}}$ and $v(y) \leq C_{\mathcal{A}}$ it holds that $fract(v(x)) \leq fract(v(y))$ iff $fract(v'(x)) \leq fract(v'(y))$ (where $fract(\cdot)$ is the fractional part);
- for each clock x with $v(x) \leq C_{\mathcal{A}}$, $fract(v(x)) = 0$ iff $fract(v'(x)) = 0$.

Let $[v]$ denote the equivalence class $\{v' \mid v \approx v'\}$. The set of equivalence classes $\{[v] \mid v \text{ is a valuation}\}$ is finite, and with V we denote its cardinality.

Known properties of the equivalence classes are summarized in the following theorem [2].

Theorem 3.2 *Let $c < C_{\mathcal{A}}$ and v and v' be two valuations; $v \in [v']$ implies that $v \models x \sim c$ iff $v' \models x \sim c$. Moreover, given a class $[v]$, the set $\{[v+t] \mid t \in \mathbb{R}^{\geq 0}\}$ is computable.*

For simplicity, we prove that we can always translate an SDMTA into an SDMTA with only one component.

Proposition 3.3 *Given an SDMTA $\mathcal{A}$ and a tuple of states $(q_1, \ldots, q_n)$, an SDMTA $\mathcal{A}'$ composed by only one $DMTA$ with a state q' can be constructed such that the tuple $(q_1, \ldots, q_n)$ is reachable by $\mathcal{A}$ iff q' is reachable by $\mathcal{A}'$.*

Proof. It is sufficient to consider the cartesian product of the sequential components of $\mathcal{A}$ and introduce two functions $pair$ and $triple$ in Ω.

Terms of the form $pair(i, \tau)$ mean that τ is in the knowledge of the i^{th} component thanks to an ϵ transition step. Terms of the form $triple(i, j, \tau)$ mean that τ is in the knowledge of the i^{th} and j^{th} components thanks to a communication between i and j.

Given a formula ϕ, with ϕ^i we denote the formula ϕ where each formula of the form $\tau \in \mathcal{K}$ is replaced with $pair(i, \tau) \in \mathcal{K} \vee \bigvee_{j \in [1,n]} triple(i, j, \tau) \vee triple(j, i, \tau)$.

In $\mathcal{A}'$ we introduce a transition $((q_1, \ldots, q, \ldots q_n), \epsilon(pair(i, \tau)), \phi^i, (q_1, \ldots, q', \ldots q_n))$ if there exists a transition $(q, \epsilon(\tau), \phi, q')$ of the i^{th} component.

In $\mathcal{A}'$ we introduce a transition $((q_1, \ldots, q, \ldots, q'', \ldots q_n), \epsilon(triple(i, j, \tau)), \phi^i \quad \wedge$

$(\phi')^j \wedge \phi, (q_1, \ldots, q', \ldots, q''', \ldots, q_n))$ if there exist two transitions (q, α_1, ϕ, q') and $(q'', \alpha_2, \phi', q''')$ of the i^{th} and j^{th} components, respectively, such that $\{\alpha_1, \alpha_2\} = \{a!(\tau), a?(\tau')\}$ and ϕ expresses the set of instances I and I' such that $I(\tau) = I'(\tau')$. □

Hence, from now on we suppose to have SDMTAs composed by only one sequential DMTA performing only ϵ transitions.

With $Term(\mathcal{A})$ we denote the set of terms τ appearing in the transitions of $\mathcal{A}$.

Given a term τ, with $Nat(\tau)$ we denote the set of natural numbers appearing in τ. If $\mathcal{K}$ is a knowledge, with $Nat(\mathcal{K})$ we denote the set $\bigcup_{\tau \in \mathcal{K}} Nat(\tau)$. Moreover, given an SDMTA $\mathcal{A}$, with $Nat(\mathcal{A})$ we denote the set $\bigcup_{\tau \in Term(\mathcal{A})} Nat(\tau)$.

Example 3.4 *If the SDMTA $\mathcal{A}$ has the transition $(q, a!(Nonce(\mu, 10)), Nonce(A, 11) \in \mathcal{K}, q')$, then $10, 11 \in Nat(\mathcal{A})$.*

A term τ *is simple for a set of message variables* Υ if it is equal to $f(\mu_{i_1}, \ldots, \mu_{i_k})$ with $\mu_{i_1}, \ldots, \mu_{i_k} \in \Upsilon$.

As an example, $f(\mu_1, \mu_2)$ is simple, and $f(\mu_1, f(\mu_2, \mu_3))$ and $f(1, \mu_2)$ are not.

With $\Psi(\Upsilon)$ we denote the set of formulae of the form $\bigwedge_\mu \mu \in \mathbb{N} \wedge \bigwedge_{\overline{\mu} \in \Upsilon} \mu \sim_{\mu, \overline{\mu}} \overline{\mu}$ with $\sim_{\mu, \overline{\mu}} \in \{=, \neq\}$.

As an example, $\psi = (\mu_1, \mu_2, \mu_3 \in \mathbb{N} \wedge \mu_1 = \mu_2 = \mu_3)$ is in $\Psi(\{\mu_1, \mu_2, \mu_3\})$.

We show now that an SDMTA can be transformed into a standard form that preserves reachability of states.

Proposition 3.5 *Given an SDMTA $\mathcal{A} = (C, \Omega, (\Sigma, X, \Upsilon, Q, q_0, \phi_0, \mathcal{K}_0, \delta))$ we can construct $\mathcal{A}' = (C', \Omega', (\Sigma, X, \Upsilon, Q', \overline{q}_0, \phi_0', \emptyset, \delta'))$ such that:*

- *$C' = \emptyset$, hence message variables can assume only natural values;*
- *$Q' = Q \cup \{\overline{q}_0, \ldots, \overline{q}_{|\mathcal{K}_0|}\}$;*
- *$\phi_0' = \bigwedge_{x \in X} x = 0 \wedge \bigwedge_{\mu \in \Upsilon} \mu \in \mathbb{N} \wedge \bigwedge_{\overline{\mu} \in \Upsilon} \mu = \overline{\mu}$, hence clocks are set to 0 at the beginning and message variables assume the same natural value;*
- *for any $(q, \epsilon(\tau), \phi, q') \in \delta'$ we have that τ is simple for Υ and ϕ is of the form $\phi_1 \wedge \phi_2 \wedge \phi_3$ where ϕ_1 is equal to $\bigwedge_{i=1}^n \tau_i \in \mathcal{K} \wedge \bigwedge_{j=1}^m \tau_j' \notin \mathcal{K}$, for some simple terms $\tau_1, \ldots, \tau_n, \tau_1', \ldots, \tau_m'$ on $\Upsilon \cup \Upsilon'$, $\phi_2 \in \Psi(\Upsilon \cup \Upsilon')$ and ϕ_3 is a formula on clocks $X \cup X'$; hence we have only simple terms and each transition expresses the relation between message variable;*
- *for any $q \in Q$, q is reachable by $\mathcal{A}$ iff q is reachable by $\mathcal{A}'$.*

Proof. First of all, if $\mathcal{K}_0 = \{\tau_1, \ldots, \tau_n\}$, then we add the transitions $(\overline{q}_n, \epsilon(), \phi_0[\mu'/\mu]_{\mu \in \Upsilon}[x'/x]_{x \in X}, q_0)$ and $(\overline{q}_i, \epsilon(\tau_i), \bigwedge_{x \in X} x = 0, \overline{q}_{i+1})$, for $i = 0, \ldots, n-1$. These transitions initialize the values of the clock and of the message variables and of the knowledge to the initial value expressed by ϕ_0 and $\mathcal{K}_0$.

Now, we delete the natural numbers in $Nat(\mathcal{A})$. If $Nat(\mathcal{A}) = \{n_1, \ldots, n_k\}$, then we consider $c_1, \ldots, c_k$ new constants and we substitute each n_i with c_i. Moreover we replace each $\mu = n_i$ with $\mu = c_i$, and, $\mu \in \mathbb{N}$ with $\mu \in \mathbb{N} \vee \mu \in \{c_1, \ldots, c_k\}$, for any $\mu \in \Upsilon \cup \Upsilon'$.

Let T be the terms appearing in the SDMTA after these operations. We can partition T in $T_1, \ldots, T_h$ such that, for any $\tau, \tau' \in T$, it holds that τ is equal to τ' but for a renaming of message variables iff $\tau, \tau' \in T_i$ for some i.

We define $\Omega' = \{f_1, \ldots, f_h\}$ and, for any $\tau \in T_i$, we replace τ with the simple term $f_i(\mu_{i_1}, \ldots, \mu_{i_k})$ if $\mu_{i_1}, \ldots, \mu_{i_k}$ is the sequence of variables occurring in τ from left to right.

Now, we replace each formula ϕ derived by the previous steps with the formula $\phi \vee \bigvee_{\psi \in \Psi(\Upsilon \cup \Upsilon')} \psi$, and, finally, we recursively replace each transition $(q, \epsilon(\tau), \phi_1 \vee \phi_2, q')$ with two transitions $(q, \epsilon(\tau), \phi_1, q')$ and $(q, \epsilon(\tau), \phi_2, q')$.

It is easy to show that the states in Q reached by $\mathcal{A}$ are the same that are reached by $\mathcal{A}'$. □

From now on, we suppose that SDMTAs are in the standard form of the proposition above.

For checking reachability for Timed Automata, in [2] they are reduced to finite state machines. We note that unbound memory cannot be simulated with a finite state machine. Hence we prove the decidability of reachability of SDMTAs by reducing the problem to the reachability problem of vector addition systems.

We recall now the model of *vector addition systems*. Given two vectors $w, w' \in \mathbb{Z}^n$, with $\pi_i(w)$ we denote the i^{th} component of w and with $w + w'$ we denote the vector w'' such that $\pi_i(w'') = \pi_i(w) + \pi_i(w')$, for any $i \in [1, n]$.

Definition 3.6 *A* vector addition system of dimension n *is a tuple $\mathcal{S} = (Q, \delta)$ such that Q is a finite set of states and $\delta \subseteq Q \times \mathbb{Z}^n \times Q$. A* configuration *is a pair (q, w) with $q \in Q$ and $w \in \mathbb{N}^n$. There exists a step from configuration (q, w) to configuration (q', w') (denoted with $(q, w) \to (q', w')$) if $(q, w'', q') \in \delta$ such that $w' = w'' + w$.*

The following theorem is proved in [8].

Theorem 3.7 *Given two configurations (q, w) and (q', w') of a vector addition system $\mathcal{S}$, the problem of checking whether there exists a sequence of steps $(q, w) \rightarrow \cdots \rightarrow (q', w')$ is decidable and EXP-SPACE hard.*

We now prove the reduction result.

Theorem 3.8 *Given an SDMTA $\mathcal{A}$ and a state $\overline{q}$ of $\mathcal{A}$, there exists a vector addition system $\mathcal{S}$ and two states q' and q'' of $\mathcal{S}$ such that $\mathcal{A}$ reaches $\overline{q}$ iff there exists a sequence of steps $(q', (0, \ldots, 0)) \rightarrow \cdots \rightarrow (q'', (0, \ldots, 0))$ of $\mathcal{S}$.*

Proof. First of all, we consider an extension of a vector addition system. If $\mathcal{S} = (Q, \delta)$ we assume $\delta \subseteq Q \times 2^{[1,n]} \times \mathbb{Z}^n \times Q$. There exists a step from the configuration (q, w) to the configuration (q', w') if $(q, In, w'', q') \in \delta$ such that $w' = w'' + w$ and $\pi_i(w) > 0$, for any $i \in In$.

It is obvious that, given an extended vector addition system we can construct an equivalent vector addition system by replacing any transition (q, In, w, q') with two transitions $(q, w', \overline{q}), (\overline{q}, w - w', q')$ where $\overline{q}$ is a new state and $\pi_i(w') = -1$ if $i \in In$ and $\pi_i(w') = 0$ otherwise.

Let $\Omega = \{f_1, \ldots, f_k\}$, Q, Υ and X be, respectively, the set of function symbols, the set of states, the set of the message variables and the set of clocks of $\mathcal{A}$.

Let $\overline{\Upsilon}$ be a set of message variables; with $\mathcal{T}(\overline{\Upsilon})$ we denote the set of terms of the form $f(A_1, \ldots, A_n)$ such that $f \in \Omega$ and $\emptyset \subset \bigcup_{i=1}^{n} A_i \subseteq \overline{\Upsilon}$. Given a set $\tilde{\Upsilon} \subseteq \overline{\Upsilon}$ and $T \subseteq \mathcal{T}(\overline{\Upsilon})$, let $T_{\tilde{\Upsilon}}$ denote the set $\{f((A_1 \cap \tilde{\Upsilon}), \ldots, (A_k \cap \tilde{\Upsilon})) \mid f(A_1, \ldots, A_k) \in T\}$.

A set $T \subseteq \mathcal{T}(\overline{\Upsilon})$ is coherent if, for any $f(A_1, \ldots, A_n), f'(A'_1, \ldots, A'_m) \in T$ and for any $i \in [1, n]$ and $j \in [1, m]$, it holds that either $A_i = A'_j$ or $A_i \cap A_j = \emptyset$.

The idea is that $f(A_1, \ldots, A_n)$ represents the instance I and the term $f(c_1, \ldots, c_n)$ such that $c_i = I(\mu)$, if $\mu \in A_i$. Hence, if $A_i = \emptyset$, then c_i is different from $I(\mu)$, for any μ. Therefore, $\mu_1, \mu_2 \in A_i$, for some i, iff $I(\mu_1) = I(\mu_2)$. Hence, a set is coherent if an instance I can be built from T.

A set $T \subseteq \mathcal{T}(\overline{\Upsilon})$ is connected if it is coherent and, for any $\tau, \tau' \in T$, $Var(\tau) \cap Var(\tau') \neq \emptyset$. The idea is that a set is connected if the values appearing in two terms are related.

Given a connected set $T \subseteq \mathcal{T}(\overline{\Upsilon})$, with $[T]$ we denote the set of connected sets T' in $T(\overline{\Upsilon})$ such that, given the sets of variables $A_1, \ldots, A_n$ appearing in T, there exist sets of variables $A'_1, \ldots, A'_n$ such that $T' = T[A'_i/A_i]_{i \in [1,n]}$.

The idea is that $[T]$ expresses the knowledge of T abstracting from the instance.

A state of $\mathcal{S}$ is a tuple $(q, [v], (A_1, T_1), \ldots, (A_m, T_m))$ such that $q \in Q$, v is a valuation on X, and $\emptyset \subset A_i \subseteq \Upsilon$ and T_i is a connected set in $\mathcal{T}(A_i)$, for any i, and $A_1, \ldots, A_m$ is a partition of Υ.

The dimension U of $\mathcal{S}$ is equal to the cardinality of $2^{T(\Upsilon)}$. Let $g : 2^{T(\Upsilon)} \rightarrow [1, U]$ be a surjective function assigning a coordinate of the vector to each subset in $T(\Upsilon)$.

The configuration $((q, [v], (A_1, T_1), \ldots, (A_m, T_m)), w)$ expresses the configuration $(q', v', I, \mathcal{K})$ of $\mathcal{A}$ such that $q = q'$, $v' \in [v]$, $(A_1, T_1), \ldots, (A_m, T_m)$ and w express the instance I and the knowledge $\mathcal{K}$.

We have a transition from the state $((q, [v], (A_1, T_1), \ldots, (A_m, T_m))$ to the state $(q', [v'], (A'_1, T'_1), \ldots, (A'_n, T'_n))$ with label (In, w) in $\mathcal{S}$ iff either $q' = q$, $[v'] = [v + t]$ for some $t \in \mathbb{R}^{\geq 0}$, $n = m$ and $A_i = A'_i$ and $T_i = T'_i$, for any $i = 1, \ldots, n$ (hence a time step is taken), or there exists a transition $(q, \epsilon(\tau), \phi_1 \wedge \phi_2 \wedge \phi_3, q')$ of $\mathcal{A}$ with $\phi_1 = \bigwedge_{i=1}^{n} \tau_i \in \mathcal{K} \wedge \bigwedge_{j=1}^{m} \tau'_j \notin \mathcal{K}$, for some simple terms $\tau_1, \ldots, \tau_n, \tau'_1, \ldots, \tau'_m$ on $\Upsilon \cup \Upsilon'$, $\phi_2 \in \Psi(\Upsilon \cup \Upsilon')$ and ϕ_3 is a formula on clocks $X \cup X'$, such that:

- $v \oplus v'$ satisfies ϕ_3;
- there exists a coherent set $T \subseteq \mathcal{T}(\Upsilon \cup \Upsilon')$ and $\tau' = f(\overline{A}_1, \ldots, \overline{A}_h)$ if $\tau = f(\mu_1, \ldots, \mu_h)$ and $\overline{A}_i$ is the set in which μ_i appears in T such that:
  - $\bigcup_{i=1}^{n} T_i = T_\Upsilon$ and $\bigcup_{i=1}^{m} T'_i = (T_{\Upsilon'})\,[\mu/\mu']_{\mu' \in \Upsilon'}$;
  - for any i, if $\tau_i = \overline{f}(\mu_1, \ldots, \mu_k)$, then there exists $\overline{f}(\overline{A}_1, \ldots, \overline{A}_k) \in T$ with $\mu_j \in A_j$, for any j;
  - for any i, if $\tau'_i = \overline{f}(\mu_1, \ldots, \mu_k)$, then for any $\overline{f}(\overline{A}_1, \ldots, \overline{A}_k) \in T$ it holds that $\mu_j \notin \overline{A}_j$, for some j;
  - for any μ_1, μ_2, it holds that $\phi_2 \Rightarrow (\mu_1 \neq \mu_2)$ iff ether $\mu_1 \in A_h$ and $\mu_2 \in A_k$ with $h \neq k$, or, there exists no $\overline{f}(\overline{A}_1, \ldots, \overline{A}_l) \in T_i$ such that $\mu_1, \mu_2 \in \overline{A}_j$ for any i and j;
- $In = \{g(T_i) \mid i = 1, \ldots, m\}$;
- if $\tau \in \bigcup_{i=1}^{n} T_i$, then $w = (0, \ldots, 0)$ and otherwise, if $H = \{i \mid Var(T_i) \cap Var(\tau) \neq \emptyset\}$, then $\pi_j(w) = \begin{cases} -1 & \text{if } g^{-1}(j) \in [T_i] \text{ with } T_i \neq \emptyset \text{ and } i \in H \\ 1 & \text{if } g^{-1}(j) \in [\{\tau'\} \cup \bigcup_{i \in H} T_i] \\ 0 & \text{otherwise} \end{cases}$.

Hence we have a transition for any step of $\mathcal{A}$. The transition updates the values of the vector by following the changes of the knowledge due to the step of $\mathcal{A}$.

Finally, we add a new state q_T to $\mathcal{S}$ and transitions with label $(In, (0, \ldots, 0))$ and target q_T from each state $((\overline{q}, [v], (A_1, T_1), \ldots, (A_m, T_m))$ with $In = \{g(T_i) \mid i = 1, \ldots, m\}$. Moreover, we add $e_1, \ldots, e_U$ transitions such that, for each i, $e_i = (q_T, \emptyset, w_i, q_T)$, where $\pi_j(w_i) = -1$

if $i = j$ and $\pi_j(w_i) = 0$ otherwise. The state q_T represents the state $\overline{q}$ of $\mathcal{A}$, and these new transitions have the purpose to set the vector to $(0, \ldots, 0)$.

It is easy to show that $\mathcal{A}$ reaches $\overline{q}$ iff there exists a sequence of steps from $((q_0, [\bigwedge_{x \in X} x = 0], \bigwedge_{\mu,\overline{\mu} \in \Upsilon} \mu = \overline{\mu}, \emptyset), (0, \ldots, 0))$ to $(q_T, (0, \ldots, 0))$. □

Hence, we have the following result.

Corollary 3.9 *Given an SDMTA $\mathcal{A}$, it is decidable in exponential time whether a state q is reachable.*

Example 3.10 *Consider an SDMTA composed by a DMTA $\mathcal{A}$ with initial condition $\phi_0 = (x = 0 \wedge \mu_1 = \mu_2)$ and a transition $e = (q, \epsilon(f(\mu_1)), x < 1 \wedge f(\mu_1) \notin \mathcal{K} \wedge \mu_1' \neq \mu_1 = \mu_2 = \mu_2', q)$.*

The set of equivalence classes are $\{x = 0, 0 < x < 1, x = 1, x > 1\}$.

As an example, from configuration $((q, x = 0, (\{\mu_1, \mu_2\}, \emptyset)), (0, \ldots, 0))$ we have a step with label $(0, \ldots, 0)$ to the configuration $((q, v, (\{\mu_1, \mu_2\}, \emptyset)), (0, \ldots, 0))$ with $v \in \{x = 0, 0 < x < 1, x = 1, x > 1\}$ representing a possible time step. The pair $(\{\mu_1, \mu_2\}, \emptyset)$ means that $\mu_1 = \mu_2$ and $f(\mu_1), f(\mu_2) \notin \mathcal{K}$.

Moreover, by using transition e we can reach the configuration $((q, x = 0, (\{\mu_1\}, \emptyset)(\{\mu_2\}, \{f(\mu_2)\})), w)$ such that w is one in position $g(f(\{\mu_1\})), g(f(\{\mu_2\})), g(f(\{\mu_1, \mu_2\}))$, and is zero otherwise. Actually, after the step we have that $\mu_1 \neq \mu_2$, $f(\mu_2) \in \mathcal{K}$ and in the future it is possible to create three different kind of instances such that:

1. *$f(\mu_1) \in \mathcal{K} \wedge f(\mu_2) \notin \mathcal{K}$ is true (since $\pi_{g(f(\{\mu_1\}))}(w) = 1$);*
2. *$f(\mu_1) \notin \mathcal{K} \wedge f(\mu_2) \in \mathcal{K}$ is true (since $\pi_{g(f(\{\mu_2\}))}(w) = 1$);*
3. *$f(\mu_1) \in \mathcal{K} \wedge \mu_1 = \mu_2$ is true (since $\pi_{g(f(\{\mu_1, \mu_2\}))}(w) = 1$);*

4 An Application to Modelling Cryptographic Protocols with SDMTAs

Security protocols, like distributed programs in general, are sensitive to the passage of time; however, most methods for the formal analysis of security properties of protocols do not take time aspects into account (see, among them, [6, 1]). The role of time in the analysis of cryptographic protocols has only recently received some attention (see [7, 9, 5]).

Time aspects can influence the flow of messages during the execution of a protocol. For instance, if a message does not arrive in a certain time interval, retransmissions or other behaviours should be considered (in this case the protocol specification should model these implementation details). Time information can be used within a protocol in order to enrich the information contained in a message (for example by using *timestamps*). Finally, the timing of the message flow may be exploited by an adversary to violate the security of the protocol.

In this section we define an instantiation of SDMTAs to the case of cryptographic protocols. We formalize an intruder's capabilities and a security property we want to verify. Then we model each principal of the protocol and the intruder with DMTAs. The execution of the protocol is modeled by the resulting SDMTA.

Definition 4.1 *A Cryptographic SDMTA (CSDMTA) is an SDMTA $\mathcal{AC} = (C, \Omega, A_1, \ldots, A_m, I)$, where C and Ω are defined as in Example 2.1, DMTAs A_i model the principals in the protocol, and I is the DMTA modelling the intruder.*

In the following, we give a formalization of a classical Dolev–Yao style intruder [6].

Definition 4.2 *Given a cryptographic protocol modelled by the CSDMTA $\mathcal{AC} = (C, \Omega, A_1, \ldots, A_m, I)$, where DMTAs $A_i = (\Sigma^i, X^i, \Upsilon^i, Q^i, q_0^i, \phi_0^i, \mathcal{K}_0^i, \delta^i)$ model the principals in the protocol, we say that $I = (\Sigma, X, \Upsilon, Q, q_0, \phi_0, \mathcal{K}_0, \delta)$ is the intruder for $\mathcal{AC}$ iff:*

- *$Q = \{q_0\}$;*
- *$\Sigma = \cup_{i=1}^m \Sigma^i$;*
- *$(q_0, \epsilon(), true, q_0) \in \delta'$;*
- *for all $i \in [1, m]$, if either $(q, a!(\tau), \phi, q') \in \delta^i$ or $(q, a?(\tau), \phi, q') \in \delta^i$, then $(q_0, a?(\tau'), true, q_0)$ and $(q_0, a!(\tau'), \tau' \in \mathcal{K}, q_0)$ are in δ, where $\tau \simeq \tau'$;*
- *for all $i \in [1, m]$, if $Pair(\tau_1, \tau_2)$ is a subformula of τ with $(q, a!(\tau), \phi, q') \in \delta^i$, then both $(q_0, \epsilon(\tau_1'), Pair(\tau_1', \tau_2') \in \mathcal{K}, q_0)$ and $(q_0, \epsilon(\tau_2'), Pair(\tau_1', \tau_2') \in \mathcal{K}, q_0)$ are in δ, where $\tau_1 \simeq \tau_1'$ and $\tau_2 \simeq \tau_2'$;*
- *for all $i \in [1, m]$, if $Pair(\tau_1, \tau_2)$ is a subformula of τ with $(q, a?(\tau), \phi, q') \in \delta^i$, then $(q_0, \epsilon(Pair(\tau_1', \tau_2')), \tau_1' \in \mathcal{K} \wedge \tau_2' \in \mathcal{K}, q_0) \in \delta$, where $\tau_1 \simeq \tau_1' \wedge \tau_2 \simeq \tau_2'$;*
- *for all $i \in [1, m]$, if $Enc(\tau_1, \tau_2)$ is a subformula of τ with $(q, a!(\tau), \phi, q') \in \delta^i$, then $(q_0, \epsilon(\tau_1'), Enc(\tau_1', \tau_2') \in \mathcal{K} \wedge \tau_2' \in \mathcal{K}, q_0) \in \delta$, where $\tau_1 \simeq \tau_1'$ and $\tau_2 \simeq \tau_2'$;*
- *for all $i \in [1, m]$, if $Enc(\tau_1, \tau_2)$ is a subformula of τ with $(q, a?(\tau), \phi, q') \in \delta^i$, then $(q_0, \epsilon(Enc(\tau_1', \tau_2')), \tau_1' \in \mathcal{K} \wedge \tau_2' \in \mathcal{K}, q_0) \in \delta$, where $\tau_1 \simeq \tau_1'$ and $\tau_2 \simeq \tau_2'$;*

With such a definition of a single state intruder, we assume that principals of the protocol use only public channels, therefore the intruder may intercept each message exchanged within the network and spread any information. This is modeled by taking Σ as the set containing all channel names appearing in the automata A_i and by adding an input transition to the intruder for any output transitions of any A_i (and an output transition for any input one). Moreover, we give the intruder classical Dolev-Yao capabilities (extraction of a term from a pair, pairing of two terms, encryption and decryption assuming the key is known). With the empty ϵ-transition the intruder may nondeterministically instantiate variables by modifying its actual instance I. With such a transition no messages are added to the memory.

4.1 Security Properties

Many security properties may be defined in order to analyze cryptographic protocols (among them, *secrecy*, *authentication*, *integrity*, *fairness*, *non-repudiation*, etc.). We give a definition of secrecy in our framework.

Intuitively, a term is secret to a principal if it never appears within its knowledge.

Definition 4.3 *Given a CSDMTA* $\mathcal{AC} = (C, \Omega, A_1, \ldots, A_m, I)$*, a term* τ *with* $Var(\tau) = \emptyset$ *and a set of principals* $P \subseteq \{A_1, \ldots, A_m, I\}$*, we say that* τ *is* P-secret *for any principal in* P *iff there exists no run* $\sigma = s^0 \rightarrow_{\alpha_1} s^1 \rightarrow_{\alpha_2} \ldots \rightarrow_{\alpha_l} s^l$ *of* $\mathcal{AC}$*, with* $s^i = (s_{A_1}, \ldots, s_{A_m}, s_I)$ *for some* $i \in [0, l]$ *and* $s_j = (q, I, v, \mathcal{K})$ *for some* $j \in P$ *such that* $\tau \in \mathcal{K}$*.*

Proposition 4.4 *Given a CSDMTA* $\mathcal{AC} = (C, \Omega, A_1, \ldots, A_m, I)$*, it is decidable whether a term* τ *with* $Var(\tau) = \emptyset$ *is* P-secret *for a set of principals* $P \subseteq \{A_1, \ldots, A_m, I\}$*.*

Proof. We build the CSDMTA $\mathcal{AC}' = (C, \Omega, A'_1, \ldots, A'_m, I')$ such that for each DMTA $\mathcal{A} \notin P$, $\mathcal{A}' = \mathcal{A}$ and for each $\mathcal{A} = (\Sigma, X, \Upsilon, Q, q_0, \phi_0, \mathcal{K}_0, \delta) \in P$, $\mathcal{A}' = (\Sigma, X, \Upsilon, Q \cup q_s, q_0, \phi_0, \mathcal{K}_0, \delta \cup \delta_s)$, where $q_s \notin Q$ and $\delta_s = \delta \cup \{(q, \epsilon(\tau), \tau \in \mathcal{K}, q_s) \,|\, q \in Q\}$. Now, for all DMTA in P we have a special transition reaching the special state q_s when the term to be kept secret appears in its knowledge. Therefore, we can say that the term τ is P-secret for $\mathcal{AC}$ if no q_s can be reached in the CSDMTA $\mathcal{AC}'$. Thus, by the decidability of state reachability given in Corollary 3.9, also P-secrecy is decidable. □

We use now the model of CSDMTAs to analyze the secrecy property for the Yahalom protocol.

4.2 The Yahalom Protocol

The Yahalom protocol [4] is designed for the distribution of a fresh symmetric key shared between two users. The protocol resorts to a trusted server, and each user is assumed to share a symmetric key with the server. Here we consider a strengthened version of the Yahalom protocol proposed by Paulson in [10]. In the standard protocol notation, we denote with $A \rightarrow B : Msg$ a message Msg sent by A and received by B, with $\{Msg\}_K$ we denote the encryption of the message Msg under the key K. If A, B and S are the principals of the protocols (with S the trusted server), Na and Nb are fresh nonces and Kas, Kbs and Kab are symmetric keys shared by the principals in the subscript, the protocol can be described as follows:

1. $A \rightarrow B \;:\; A, Na$
2. $B \rightarrow S \;:\; B, Nb, \{A, Na\}_{Kbs}$
3. $S \rightarrow A \;:\; Nb, \{B, Kab, Na\}_{Kas}, \{A, B, Kab, Nb\}_{Kbs}$
4. $A \rightarrow B \;:\; \{A, B, Kab, Nb\}_{Kbs}, \{Nb\}_{Kab}$

It is also assumed that principal A only knows elements in the set $\{A, B, S, Kas\}$, principal B knowledge is given by $\{B, S, Kbs\}$, and the trusted server S knows $\{A, B, Kas, Kbs\}$.

User A starts the protocol by communicating to B its intention to share a new session key with it (step 1). User B generates a fresh nonce Nb and creates a new term containing the identity of A and its nonce Na encrypted with the key Kbs shared with the trusted server. User B sends its identity, its nonce Nb and the encrypted term to the server (step 2). Server S deciphers the encrypted term, obtains the identity of A and generates a new fresh key Kab. It also builds two terms encrypted with Kas and Kbs, and sends the whole message to A (step 3). Finally, A deciphers the first encryption and checks whether it contains nonce Na. If this is the case, A sends to B the second term encrypted with Kbs and mutually authenticates to B by sending nonce Nb encrypted with the fresh session key (step 4).

The basic requirement that this protocol must satisfy is the secrecy of the key Kab (in every session, the value of Kab must be known only by the participants playing the roles of A, B and S).

The protocol can be adapted to take timeouts and retransmissions into account. In step 1, after sending its message, A may start a timer while waiting for the message of step 3. If the timeout occurs, A may retransmit its message (the same nonce Na can be sent as it was already sent in clear). As some amount of time must pass while B and S receive and elaborate messages, if A receives an answer too early, this might signal some misbehaviour. Suppose that A knows the encryption and decryption times of B and S, we might adapt the Yahalom protocol to take time into account. In a formalism where time aspects are not considered this would not be possible.

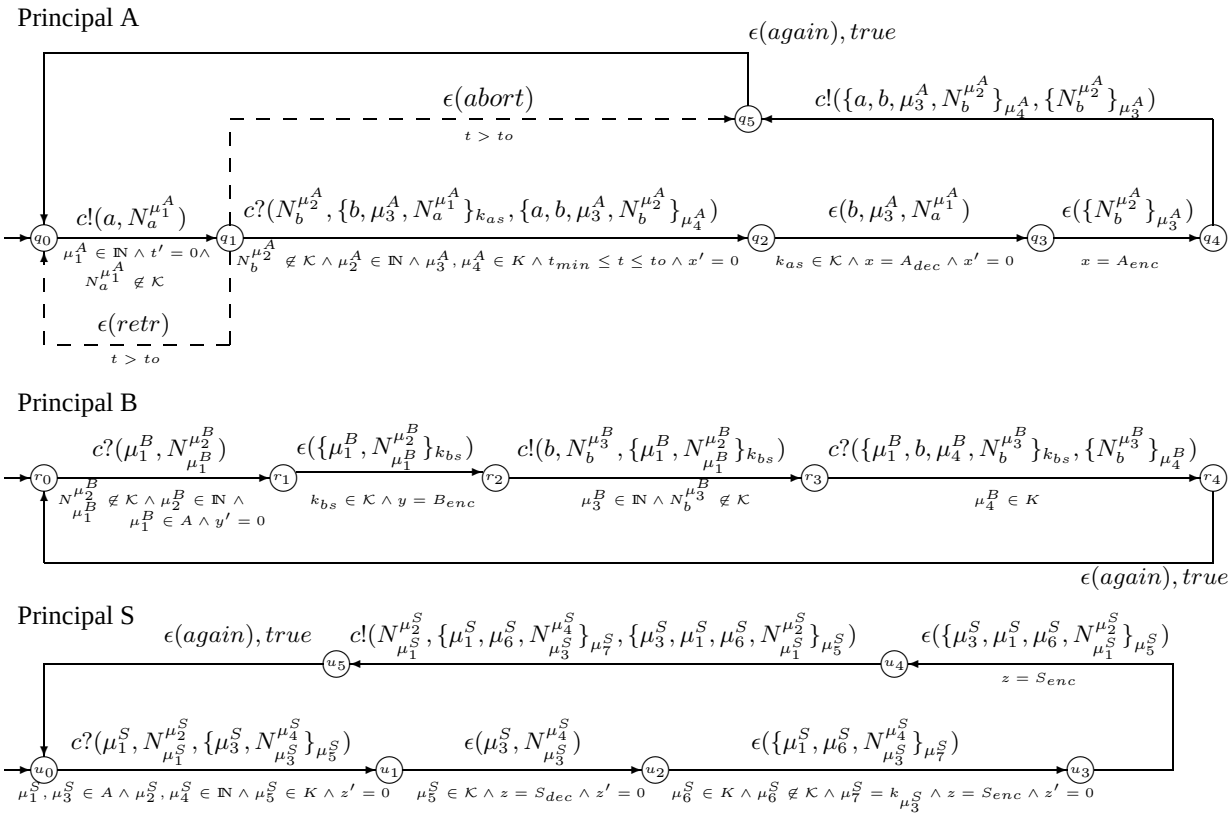

Figure 2. Specification with CDMTA of the principals of the Yahalom protocol.

4.3 Specification of the Yahalom Protocol through SDMTAs

We model the execution of the Yahalom protocol through a CSDMTA $\mathcal{AC} = (C, \Omega, A, B, S, I)$, where A, B and S are the DMTAs representing the principal A, the principal B and the trusted server S, respectively, while I is the intruder as specified in Definition 4.2.

The principals in the protocol are specified by the following DMTAs, graphically represented in Figure 2:

$$A = (\{c\}, \{x, t\}, \Upsilon^A, Q^A, q_0, \phi_0^A, \{a, b, s, k_{as}\}, \delta^A);$$
$$B = (\{c\}, \{y\}, \Upsilon^B, Q^B, r_0, \phi_0^B, \{b, s, k_{bs}\}, \delta^B);$$
$$S = (\{c\}, \{z\}, \Upsilon^S, Q^S, u_0, \phi_0^S, \{a, b, s, k_{as}, k_{bs}\}, \delta^S).$$

We assume that all communications happen through the only public channel c. All initial formulas ϕ_0^i reset to 0 the clocks of the DMTA i and set message variables to a random value. All principals $C \in \{A, B, S\}$ have the set of variable messages $\Upsilon^C = \{\mu_1^C, \ldots, \mu_6^C\}$.

For readability, in Figure 2 we use the notation $N_\mu^{\mu'}$ for $Nonce(\mu, \mu')$; τ, τ' for $Pair(\tau, \tau')$ and $\{\tau\}_\mu$ for $Enc(\tau, \mu)$. Moreover, we suppose that when a message $(\tau_1, \ldots, \tau_n)$ is send/received, $\tau_1, \ldots, \tau_n$ fall in the knowledge (this is simply implementable by a finite sequence of ϵ transitions).

In the figure, for all principals $i \in \{A, B, S\}$, we omit the condition $\forall \mu \in \Upsilon^i \, \mu' = \mu$ from transition guards, but we assume that it holds for all transitions but the transitions with label $\epsilon(again), true$, where $true$ means that message variables assume a value non-deterministically.

Moreover, $\mathcal{K}$ on a transition is intended to refer to the knowledge in the configuration to which the source state of the transition belongs.

The guarantee that each message sent within the network will be received by the right participant (without considering the intruder which is capable of intercepting all messages) is given by the reducibility of the terms sent.

The protocol is started by A that sends a message $a, N_a^{\mu_1^A}$, where $N_a^{\mu_1^A}$ is a fresh unused nonce, and starts waiting for a reply from S. In the formula on this transition, $N_a^{\mu_1^A}$ should not be in the knowledge of A. When A sends the message to B it also resets a timer t to 0, and a timeout can be fixed (in the figure it is fixed to the value to). Moreover, we assume that A knows the encryption/decryption

time of B and S, and hence it does not expect to receive a message before time t_{min}.

After B receives the message from A, it checks whether the nonce is unused and generates a fresh nonce, and builds the encrypted terms it will send to S. Note that B_{enc} is a constant representing the time needed by B for encrypting a term.

When S receives the term from B it deciphers the encrypted term and builds the ciphered terms it should send to A. Again, S_{dec} and S_{enc} are constant values representing the time needed by S to decipher and cipher a term.

Note that the key chosen by S through the instantiation of the variable μ_6^S is fresh (since it does not appear in the knowledge of S, it was not chosen before).

When S finishes to elaborate the messages, it sends to A the message containing the new fresh key in the two subterms ciphered with k_{as} and k_{bs}, respectively.

If A receives such a message before the timeout has expired, it extracts the new fresh key and uses it to cipher the nonce sent by B to S, and then sends to B the ciphered nonce together with the term containing the fresh key built by S for B.

Some policies can be implemented when A does not receive an answer before the timeout expires. In Figure 2, we considered a couple of them. In one case, A may think that B or S are "down" and abort the execution of the protocol (dashed transition to state q_5). In the other case, A may think that its first message was lost, and retransmits its first message (dashed transition to state q_0).

Given the specification of the Yahalom protocol through the model of CSDMTA, we can now prove the secrecy for the fresh generated key Kab (the intruder should not be able to deduce any information about the key). In particular we require that after the generation of the key (state u_3 of S) the key $I(\mu_6^S)$ is $\{I\}$-secret for the CSDMTA $\mathcal{AC}$ modeling the execution of the protocol. Such a property can be easily verified through the rechability proof method we have shown in Example 3.10.

Other security properties of cryptographic protocols may be specified within our framework. In the case of the Yahalom protocol, for example, another requirement is that the principal A must be properly authenticated to B. Intuitively, a principal A truly authenticates to a principal B if whenever B thinks of communicating with A, it is really communicating with A. This can be specified and verified by resorting to a simple temporal logic over the runs of the SDMTA, and by requiring that whenever the principal B ends a run of a communication protocol with A, then, in the same run, A has previously started the communication protocol.

5 Conclusions and Future Works

We have defined systems of communicating automata endowed with structures to store information and functions to elaborate them. We have given a construction of a finite region graph and hence proved decidability of the reachability. We have shown an application to describe and verify cryptographic protocols.

As a future work we are interested in an application to security in distributed data bases. We plan also to enrich the formalism with probability and to develop a concept of bisimulation for this version of SDTMAs and give an algorithm for decision. Moreover, we want to study the expressive power of our model. We believe that when SDMTAs are endowed with a concept of recognized language, non context–free languages such as $\{a^n b^m c^k \mid n \geq m \geq k \geq 1\}$ are recognized while context–free languages such as $\{a^n b^n \mid n \geq 1\}$ are not.

References

[1] M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: The Spi calculus. In *4th ACM Conference on Computer and Communications Security*, pages 36–47. ACM Press, 1997.

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

[3] T. Amnell, G. Behrmann, J. Bengtsson, P. R. D'Argenio, A. David, A. Fehnker, T. Hune, B. Jeannet, K. G. Larsen, M. O. Moeller, P. Pettersson, C. Weise, and W.Yi. Uppaal-now, next and future. *Springer LNCS*, 2067:99–124, 2000.

[4] M. Burrows, M. Abadi, and R. Needham. A logic for authentication. Technical Report 39, Digital Systems Research Center, Feb 1989.

[5] R. Corin, S. Etalle, P. H. Hartel, and A. Mader. Timed analysis of security protocols. Technical Report to appear, Centre for Telematics and Information Technology, Univ. of Twente, The Netherlands, March 2005.

[6] D. Dolev and A. C.-C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–207, 1983.

[7] R. Gorrieri, E. Locatelli, and F. Martinelli. A simple language for real-time cryptographic protocol analysis. In *12th European Symposium on Programming (ESOP03)*, volume 2618 of *LNCS*, pages 114–128. Springer, 2003.

[8] S. R. Kosaraju. Decidability of reachability in vector addition systems. In *14th Annual ACM Symp. on Theory of Computing*, pages 267–281. ACM Press, 1982.

[9] M. Napoli, M. Parente, and A. Peron. Specification and verification of protocols with time constraints. *Electronic Notes in Theoretical Computer Science*, 99:205–227, 2004.

[10] L. C. Paulson. Relations between secrets: Two formal analyses of the Yahalom protocol. *Journal of Computer Security*, 9(3):197–216, 2001.

[11] S. Yovine. Kronos: A verification tool for real-time systems. *International Journal on Software Tools for Technology Transfer*, 1:123–133, 1997.