Formal Specification and Verification of Timed Communicating Systems

S.Akshay

Master's report, M2, MPRI

under the guidance of Paul Gastin *and* Benedikt Bollig

LSV, ENS Cachan

Contents

1 Introduction

In formal language theory the most basic and important concept is that of words over a finite alphabet. Words are used to describe system executions, runs, etc. And we also have a canonical automaton model of a finite state machine and we can talk about satisfaction of logical formula over words. One of the most famous connections between automata theory and classical logic was established by Büchi and Elgot early in the sixties [Büc60, Elg61]. They showed that the language of a finite automaton can be expressed by a formula from monadic second-order logic (MSO) and that, conversely, any MSO formula can be effectively transformed into an equivalent finite automaton. Such a study of the relation between logical formalisms and operational automata has been a fascinating area of theoretical computer science. This has also led to many attempts at generalizing the theory including trying to extend and abstract the definition of words themselves.

One natural approach is to look at a timed extension. This is not only a natural extension but also a very important one especially for verification of safety critical timed systems. To do this formally we look at timed words which are nothing but words tagged with a real number to show the time of occurrence. The corresponding automaton characterization is the timed automata model [AD94]. However their languages are not closed under complementation and their language inclusion problem is undecidable in general. Hence various restricted classes have been looked at with their corresponding logical characterizations. One such restriction is the event clock automaton [AFH99] which has implicit clocks allowing it to record or predict time elapses. This is well-suited for real-time specifications (such as bounded response time) and allows a suitable logical characterization [D'S03] by monadic second order logic (MSO) over timed words.

Another approach is to look at words as total orders and then abstract this concept by allowing partial orders. Further if we add a labelling alphabet to it we get another notion that has proved to be very useful in system and design, i.e, the notion of message sequence charts (MSC). MSCs have been known for a long time independently as they serve as informal documentation of design requirements that are referred throughout the design process and even in the final system integration and acceptance testing [ITU98, ITU99].

One of the convincing reasons for use of these MSCs is the advantage they have in describing behaviour of automata models which require partial order methods. In particular MSCs are used for describing Communicating Finite-state Machines (CFMs), which are a fundamental model for concurrent systems and communicating protocols. These CFMs have communicating channels between the constituent finite state automata and a single MSC diagram subsumes a whole set of sequential runs of the CFM. An early approach to using these MSCs considered only "universally bounded" MSCs

which means the existence universal bounds on the channels of the CFMs. This constraint in fact turns a CFM into a finite state device. Thus over such universally bounded MSCs, automata (CFMs), logic (monadic second order over MSCs) are all equivalent [HMK$^+$05]. This was further generalized by Genest et al. [GKM06] in a recent work where they gave such a Büchi-like characterization i.e, an equivalence in terms of both logical and automaton characterizations, with a more relaxed definition of channel bounds. They in fact consider "existential bounds" on channels which intuitively means that there should the existence of bounds on some ideal run of the MSC but no restriction on channel size in general.

Our goal in this report is to try to merge the above mentioned approaches at extending and abstracting words. We aim to study the interaction between partial orders and timing constraints. Thus we would like to define notions of timed partial orders and to get a logical and an equivalent automata theoretic characterization for this as in the case of untimed MSCs. Further we would of course like to look at implementable architectures for doing this. Past approaches to introducing timing constraints in such MSCs with a formal semantics and analysis have been looked in [AHP96, BAL97].

Outline. We start with preliminaries in Section 2, move on to the definition of an MSC and some basic ideas about MSCs in Section 3. Section 4 comprises of an attempt to introduce timing into the MSCs. Then we have the logical characterization in Section 5 followed by the automaton characterization in Section 6. We combine the three above sections to get an equivalence result in Section 7. In Section 8 we look at another definition for introducing timing in MSCs which is more natural for specification but less robust than our previous definition. In Section 9 we look at some preliminary implementability issues for our automata models. Finally we conclude with future directions and comments in Section 10.

2 Preliminaries

In this section we introduce or rather recall some basic definitions as well as fix some notations that we will be using in this report.

An *alphabet* is a finite set. Its elements are called *symbols* or *letters*. A *word* is a finite sequence of symbols juxtaposed, where we denote the empty string containing no symbols by ϵ. A *language* is a set of strings from some one alphabet. If Σ is some fixed alphabet, the set of all strings over Σ is a language denoted by Σ^*.

A *(binary) relation* R on set S is a set of pairs of elements of S, i.e, a subset of $S \times S$. We write "$a\ R\ b$" to denote that (a, b) is a pair in R. The relation R is said to be a *partial order* if R is *reflexive* ($a\ R\ a$ for all a in S), *antisymmetric* ($a\ R\ b$, $b\ R\ a$ implies $a = b$ for all a, b in S) and *transitive* ($a\ R\ b$, $b\ R\ c$ implies $a\ R\ c$ for all a, b, c in S). It is called a *total order* if

in addition for any pair of elements $a, b \in S$ either the pair (a, b) or (b, a) belongs to R. A *linear extension* of a partial order R is a total order R' which contains all the pairs of R. Note that if $S = \{s_1, ..., s_n\}$ is a finite set and R is a total order then we can represent S as a sequence $(s_{i_1}, ..., s_{i_n})$ such that $s_{i_1}\ R\ s_{i_2}\ ...\ s_{i_{n-1}}\ R\ s_{i_n}$.

For an alphabet Σ, a Σ-*labelled partial order* is a triple $(E, \preceq, \lambda)$ where E is a set, $\preceq$ is a partial order on E called its *ordering relation* and $\lambda : E \to \Sigma$ is a labelling function. A *linearization* of a labelled partial order $(E, \preceq, \lambda)$ is any labelled partial order over the same set of events, labelling function and alphabet but whose ordering relation is a linear extension of $\preceq$. Then E is a totally ordered set and can be expressed as a sequence $(e_1, ..., e_n)$ with $e_1 \preceq' e_2 \preceq' ... \preceq' e_{n-1} \preceq' e_n$. We use $\prec$ to denote the set of all pairs of $\preceq$ that are not of the form (e, e) for any $e \in E$. Also for letter a of alphabet Σ, $\lambda^{-1}(a)$ refers to the set of all elements of E which map to a under λ. And similarly, for a set Σ' subset of Σ, $\lambda^{-1}(\Sigma')$ refers to the set of all elements of E that map under λ to some element of Σ'.

The set of real numbers is denoted $\mathbb{R}$ and the set of non-negative reals is $\mathbb{R}^{\geq 0}$. $\mathbb{N}$ is the set of natural numbers. Then, a *timed word* over alphabet Σ is a member σ of $(\Sigma \times \mathbb{R}^{\geq 0})^*$ such that if $\sigma = (a_0, t_0)(a_1, t_1)...$, then $t_i \leq t_{i+1} \forall i \in \mathbb{N}$. This is the basic model of timed extension that we have and all other models will be abstractions of this, just as partial orders and MSCs are abstractions of words.

We will finish this section by recalling a definition of a labelled transition system which will be used to define our automaton model. A *finite labelled transition system* over a set of *labels* Λ is a triple $(S, \rightarrow, \iota)$, where S is a finite set of *states*, $\iota \in S$ is a special *initial state* and $\rightarrow$ defined to be a finite subset of $S \times \Lambda \times S$ is the *transition relation*. If $s, s' \in S$, $a \in \Lambda$ and $(s, a, s') \in \rightarrow$ then we write $s \xrightarrow{a} s'$ by which we mean intuitively that at state s the transition system reads label a and makes a transition to reach state s'.

3 Message Sequence Charts

Message sequence charts are a popular visual formalism for documenting design requirements for concurrent systems. These are widely used in industry and have been standardized in the ITU norm Z.120 [ITU98, ITU99]. The advantage of using this for reasoning about concurrent communicating systems is both succinctness and simplicity of comprehension.

An example of an MSC is shown in Figure 1. Vertical lines in the chart correspond to asynchronous processes or agents which start from the top of the line and finish the execution at the bottom thus denoting the passage of time in between. Messages exchanged between these processes are described by arrows where the tail is the sending event and the head is the receiving

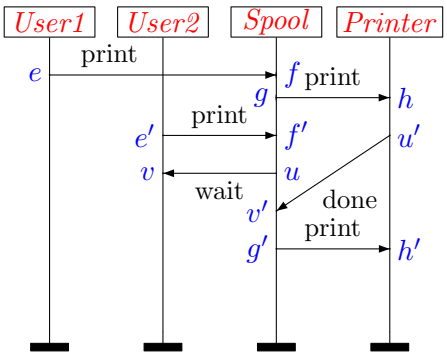

Figure 1: A Message Sequence Chart

event. Clearly arrows can be drawn horizontally or sloping downwards but not upwards.

To formalize this notion we start by fixing a finite and non-empty set of *Agents* or *processes*, Ag, with at least two elements. Agents will be denoted $p, q, ...$ They act by either sending a message, denoted $p!q$, meaning that process p sends message to process q or by receiving a message, denoted $q?p$, meaning that process q receives from process p. Each send or receive in a process occurs at an *event*. We do not have message contents or local events, i.e, events that neither send or receive, in what follows but with a little work this can be incorporated in this approach. Thus we define the set of *communication actions* as $Act = \{p!q \mid p, q \in Ag, p \neq q\} \cup \{p?q \mid p, q \in Ag, p \neq q\}$. The set of *actions* on a single process p is defined as $Act_p = \{p!q \mid q \in Ag \backslash \{p\}\} \cup \{p?q \mid q \in Ag \backslash \{p\}\}$. Note that $\bigcup_{p \in Ag} Act_p = Act$. This set Act will form the alphabet of the MSC.

Next we fix a finite set E and a partial order $\preceq$ defined on the set of events called the *visual order*. The elements of E are the *events* and they are denoted $e, e' ...$. We also introduce a *labelling function* λ which maps each event to a communication action. Then such a triple $(E, \preceq, \lambda)$ is an Act-labelled partial order. We will use the following notations.

- $\downarrow e = \{e' \in E \mid e' \preceq e\}$.
- Events of the p^{th} process. $E_p = \{e \in E \mid \lambda(e) \in Act_p\}$.
- Restriction to p^{th} process: $\prec_p = \{(e, e') \in \prec \mid e \text{ and } e' \in E_p\}$
- Immediate predecessor on the p^{th} process: $\prec\!\!\cdot_p = \prec_p \backslash (\prec_p)^2$.
- The set of send events is denoted by $S = \{e \in E \mid \exists p, q \in Ag : \lambda(e) = p!q\}$. $R = E \backslash S$ is the set of receive events.
- The message relation Msg is a function that assigns to each send event its corresponding receive. Formally we define,

$Msg : S \to R$,
$e \mapsto e'$ s.t $\lambda(e) = p!q$, $\lambda(e') = q?p$, and
$|\{e'' \mid \lambda(e'') = p!q, e'' \prec_p e\}| = |\{e'' \mid \lambda(e'') = q?p, e'' \prec_q e'\}|$.

- $Ag(e) = p \in Ag$ s.t $\lambda(e) \in Act_p$.

With these notations we have the formal definition of the MSC,

Definition 1 (MSC) *A* message sequence chart M *over* Ag *is an* Act*-labelled partial order* $(E, \preceq, \lambda)$ *such that,*

1. *each process is total :* $\lambda^{-1}(Act_p)$ *is totally ordered by* $\preceq$
2. *completeness : Messages preserve the partial order, i.e* $\{(e, Msg(e)) \mid e \in S\} \subseteq \preceq$ *and the number of events labelled with* $p!q$ *is equal to the number of events labelled with* $q?p$ *for all* $p, q \in Ag$*, i.e.,* $|\lambda^{-1}(p!q)| = |\lambda^{-1}(q?p)|$.
3. *fifo behaviour: each channel is required to have first-in-first-out queue behaviour. Formally, for all* $s_1, s_2 \in S$ *if* $s_1 \prec_p s_2$ *for some* p*, and if* $\lambda(s_1) = \lambda(s_2) = p!q$*, then* $Msg(s_1) \prec_q Msg(s_2)$.

Semantics as words :

Intuitively, since any MSC is just a partial order we can think of linearizing it in different ways to get different total orders. And since we have a labelling on the set of events each total order corresponds to a word over the alphabet of labels and thus we have a set of words corresponding to an MSC. An MSC and its possible set of linearizations in the form of words is shown pictorially in Figure 2.

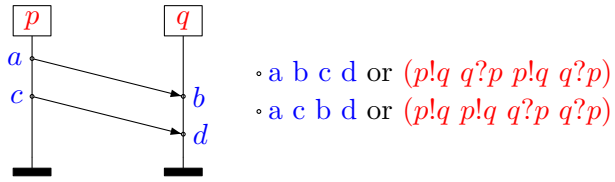

Figure 2: An MSC and its two possible realizations

Further, for any linearization of a given MSC $(E, \preceq, \lambda)$ with its set of events $E = \{e_1, ..., e_n\}$, we have a total order of events $e_{i_1} \preceq ... \preceq e_{i_n}$ with $\{i_1, ..., i_n\}$ being a permutation of $\{1, ..., n\}$. This can be represented as a word over the alphabet Act, $(\lambda(e_1)\lambda(e_2)...\lambda(e_n))$. Thus the set of linearizations of an MSC corresponds to and can be identified with a subset of Act^*.

Definition 2 (Extended MSCs) *Let Σ be an alphabet. Then a* Σ-extended MSC *over Ag is a structure $(E, \preceq, \lambda, \rho)$ such that $(E, \preceq, \lambda)$ is an MSC over Ag and $\rho : E \to \Sigma$. But in fact by a slight abuse of notation we will identify such a structure with a triple $(E, \preceq, \lambda')$ where $\lambda' : E \to Act \times \Sigma$, $\lambda'(e) \mapsto (\lambda(e), \rho(e))$. Let the set of all such extended MSCs be denoted $\mathbb{MSC}(Act \times \Sigma)$.*

Now we look at the restrictions on channel size. As mentioned in the introduction we need some such restriction to get an equivalence between logic and automata formalisms. For this first let B be a positive integer. A word $w \in Act^*$ is said to be B-bounded if for any prefix u of w and any $p, q \in Ag$, the number of occurrences of $p!q$ exceeds the number of occurrences of $q?p$ by at most B. An MSC M is said to be *existentially B-bounded* or $\exists$-B-bounded if it has some B-bounded linearization. It is called *universally B-bounded* or $\forall$-B-bounded if all its linearizations are B-bounded. For example the MSC in Figure 3(a) is $\forall$-2-bounded since at any time of the execution there are at most 2 messages in each channel. The MSC in Figure 3(b) is not $\forall$-1-bounded since during execution the process p could send two messages before q receives even one, i.e, there exists a linearization say $(p!q, p!q, p!q, p!q, q?p, q?p, q?p, q?p)$ for which the channel contains during execution more than one message. However there exists a linearization, namely $(p!q, q?p, p!q, q?p, p?q, q?p, p!q, q?p)$ which is 1 bounded, hence the MSC is $\exists$-1-bounded.

(a) **(b)**

Figure 3: A $\forall$-2-bounded and a $\exists$-1-bounded MSC

A set of MSCs is said to be $\exists$-B-bounded (respy $\forall$-B-bounded) if each MSC in the set is $\exists$-B-bounded (respy $\forall$-B-bounded). Further such a set is called existentially bounded (respy universally bounded) if there exists a B such that it is $\exists$-B-bounded (respy $\forall$-B-bounded). An extended-MSC $(E, \preceq, \lambda, \rho)$ is said to be $\exists$(respy $\forall$)-B-bounded if its projection, i.e., the MSC $(E, \preceq, \lambda)$ is existentially (universally) B-bounded.

4 Timing the MSC

In this section we will explain our first (natural) approach at attaching timing information to the message sequence charts. This is done by attaching a time stamp to each event of an MSC to get a time stamped or a timed MSC. The time stamps are of course non-negative real numbers. Such a timing is very realistic since this can be seen as modeling the real-time execution of each process. This does not however give a full linearization of the MSC since two events might occur at the same time on two different processes. This is shown in Figure 4. This property of being a real-time execution or "realization" makes this timing approach an ideal formalism to describe the behaviours and realizations of automata or logics with timing information. We will demonstrate this idea in this and the following three sections.

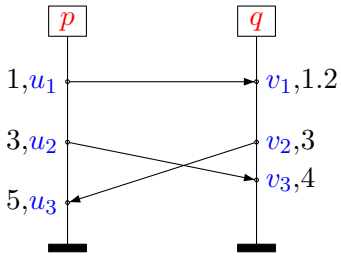

Figure 4: A Timed Message Sequence Chart

Definition 3 (T-MSC) *A* Timed-MSC *or* Time-stamped MSC *over Ag is a tuple* $(E, \preceq, \lambda, t)$*, where* $(E, \preceq, \lambda)$ *is an MSC over Ag and* $t : E \to \mathbb{R}^{\geq 0}$ *is a function that assigns a non-negative real number to each event such that if* $e_1 \preceq e_2$ *then* $t(e_1) \leq t(e_2)$*. The set of all T-MSCs over Ag is denoted* $\mathbb{TMSC}$*.*

Semantics as timed words: As in the untimed case, if $E = \{e_1, ..., e_n\}$, any linearization of the labelled partial order, can be written as a sequence $\sigma = e_1 e_2 ... e_n$ without loss of generality, where the events are related by the total order of the linearization. Now, any such linearization σ is said to be a realization of the T-MSC if $t(e_i) \leq t(e_{i+1}), \forall i$. Now any such realization can be represented as a timed word $(\lambda(e_1), t(e_1))\ (\lambda(e_2), t(e_2))\ ...\ (\lambda(e_n), t(e_n))$ over Σ. Thus the set of realizations of a T-MSC is a timed language.

Hereafter we may ignore writing the alphabet Ag for MSCs and T-MSCs if it is clear from the situation which alphabet we are referring to. We remark here that T-MSCs can be seen as monotonicity-preserving $\mathbb{R}$-extended MSCs assuming we can extend the notion of extended MSCs to those over $\mathbb{R}$ which is an infinite set.

4.1 Existential and universally bounded T-MSCs

As in the untimed case if we eventually hope to get an equivalence between logic and automata over such T-MSCs we need a notion of bounded T-MSCs. So, we extend the definition of existential and universal bounds from MSCs to T-MSCs. A T-MSC $(E, \preceq, \lambda, t)$ is said to be *untimed-existentially-B-bounded* denoted $\exists^u$-B-bounded if its projection, the MSC $(E, \preceq, \lambda)$ is existentially-B-bounded. The set of all $\exists^u$-B-bounded T-MSCs over Ag is denoted $\mathbb{TMSC}^B$. We have similar definitions for untimed-universally-B-bounded T-MSCs.

In fact such a definition of directly lifting the definition of bounds from the untimed version is not totally intuitive. In particular we could have an example such as Figure 5. The two T-MSCs have the same underlying MSC as Figure 3(b) and hence they are both $\exists^u$-1-bounded. However we observe that for the T-MSC in Figure 5(b) there is no realization which preserves the channel bound of 1 during its execution. Nevertheless we use this definition of existential bound on T-MSCs since we believe that with some work it is possible to extend it to cover the intuition. Currently we use this definition only for purposes of proving forth-coming theorems, so we stick to it.

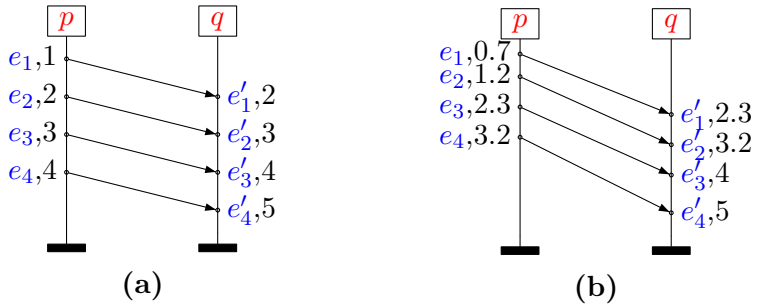

Figure 5: Two different $\exists^u$-1-bounded T-MSCs

We also remark here that for $\forall$-bounded MSCs the situation is the contrary. That is, there exist T-MSCs which are not $\forall^u$-B-bounded but all of whose realizations as timed words are B-bounded. This can occur since the underlying MSC might have some linearizations which are not B-bounded but are not realizable because of the timing constraints.

4.2 Timing alphabet

Now we come to a very important concept of the timing alphabet. The timing alphabet as we define it below is what determines which events get timed. In effect this means that this allows us to customize the timing to suit the needs of the user.

Let $M = (E, \preceq, \lambda)$ be an MSC over Ag. We fix a finite set of symbols Δ. Each $\alpha \in \Delta$ is interpreted for the MSC M as a partial function $\alpha^M : E \to E$

with $\alpha^M(e) \neq e$ for all $e \in E$. This induces the relation $\prec_\alpha = \{(e, e') \in \prec \mid$ either $e' = \alpha^M(e)$ or $e = \alpha^M(e')\}$. We call Δ a timing alphabet.

We also extend this notion to timed MSCs and extended MSCs naturally and have correspondingly α^T for TMSC T over Ag and $\alpha^{M'}$ for some Σ-extended MSC M' over Ag. But then we note that if two TMSCs or extended MSCs have the same underlying MSC then their timing alphabet is already defined and so is the map. In other words if $M = (E, \prec, \lambda)$ is an MSC over Ag, $T = (E, \preceq, \lambda, t)$ is a T-MSC over Ag and $M' = (E, \preceq, \lambda, \rho)$ is a Σ-extended MSC over Ag, then $\alpha^M = \alpha^T = \alpha^{M'}$.

The usefulness of this model and the practical implementability of the corresponding automata model depends vitally on the choice of these symbols and their definition. In other words, for usability we can restrict the events we want to be timed by defining the function α^M according to our needs and requirements. For instance one set of symbols we could use is the so-called "Last occurrence" symbols which corresponds to the event-recording automata as defined by Alur et al. [AFH99]. We give a more concrete example below.

Example 4 *We will fix* $\Delta_{tc} = \bigcup_{a \in Act} \{Prev_a, Next_a\} \cup \{Msg\}$*, where the symbols are interpreted as follows over an MSC* $M = (E, \preceq, \lambda)$*.*

- $Msg^M(e) = Msg(e)$ *for all events e, where Msg is the message relation already defined.*
- *The last appearance relation of an action* $a \in Act$ *is defined as* $Prev_a \in \Delta$*, such that* $Prev_a^M(e) = e' \in E$ *s.t* $\lambda(e') = a, e' \prec e$ *and* $\nexists\ e'' \in E$ *with* $\lambda(e'') = a$ *and* $e' \prec e'' \prec e$*. If such an* e' *does not exist then* $Prec_a^M(e)$ *is not defined.*
- *The next appearance relation is defined similarly as* $Next_a \in \Delta$*, for* $a \in Act$ *such that* $Next_a^M(e) = e' \in E$ *s.t* $\lambda(e') = a, e \prec e'$ *and* $\nexists\ e'' \in E$ *with* $\lambda(e'') = a$ *and* $e \prec e'' \prec e'$*. If such an* e' *does not exist, then* $Next_a^M(e)$ *is not defined.*

With the above example we are actually able to see that this approach generalizes the approach of D'Souza [D'S03] in the timed words case.

4.3 Intervals and interval alphabets

To specify timing constraints which will be used in the logical and automata formalizations we use rational bounded intervals over the real line. These can be open or closed intervals but we require them to be non-empty and the bounds to be rational. We also will use the symbol $[\perp, \perp]$ as a special interval which comprises of the single "undefined" symbol $\perp$ not contained in $\mathbb{R}$.The set of all intervals is denoted by $\mathcal{I}$. Now we define an *interval alphabet*

based on Σ to be a finite non-empty subset of $\Sigma \times \mathcal{I}^{\Delta}$. Thus an element of an interval alphabet is of the form (a, g) with $a \in \Sigma$ and $g : \Delta \to \mathcal{I}$.

4.4 Interval alphabet extended MSCs and TMSC realizations

We look at extended-MSCs whose alphabets are interval alphabets based on Ag as defined above. In particular we consider a $\mathcal{I}^{\Delta}$-extended MSC $M = (E, \preceq, \lambda, (g_e)_{e \in E})$ where from Definition 2 it follows that $(E, \preceq, \lambda)$ is an MSC over Ag and $(g_e)_{e \in E} : E \to \mathcal{I}^{\Delta}$ is a map that associates to every event e another map from Δ to $\mathcal{I}$ i.e, $g_e : \Delta \to \mathcal{I}$.

Definition 5 (well-extended MSCs) *We call an $\mathcal{I}^{\Delta}$-extended MSC $(E, \preceq, \lambda, (g_e)_{e \in E})$ a* well-extended MSC *if it satisfies the following property: for any event e and any $\alpha \in \Delta$, $g_e(\alpha) = [\perp, \perp]$ if and only if $\alpha^M(e)$ is not defined. The set of all such well-extended MSCs is denoted by $\mathbb{WMSC}$.*

The idea here is that the intervals in the extended alphabet correspond to the timing constraints of pairs of events as defined by the timing alphabet. Further the well-extended MSC definition is motivated by the fact that we don't want to have intervals in the extended alphabet corresponding to invalid timing constraints. We illustrate these ideas in the following simple example.

Example 6 *To avoid confusion lets assume that the timing alphabet comprises only of message constraints, i.e, $\Delta = \{Msg\}$. Then let us look at the simple $\mathcal{I}^{\Delta}$-extended MSC M of Figure 6 with just two events e, e' on two processes p, q and a message between them. $M = (E, \preceq, \lambda, (g_e)_{e \in E})$ where $E = \{e, e'\}$, $\lambda(e) = p!q, \lambda(e') = q?p$ and $g_e(Msg) = (0, 2)$. $g_{e'}(Msg) = [\perp, \perp]$. Then M so defined is a well-extended MSC since at event e', $Msg^M(e')$ is not defined and this corresponds to labelling e' by $[\perp, \perp]$. Conversely in e where $Msg^M(e) = e'$, the labelling interval is not $[\perp, \perp]$.*

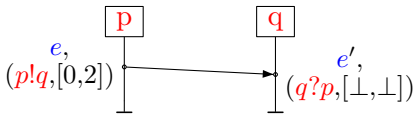

Figure 6: well-extended MSC M

Thus we have a concept of an MSC extended with "nice" timing intervals and we also have time-stamped MSCs. So the natural question to ask would be concerning the relation between these two concepts. In fact it is easy to observe that any well-extended MSC can be seen as describing a set of possible T-MSCs. More formally we have,

Definition 7 *A T-MSC $T = (E, \preceq, \lambda, t)$ is said to be a* realization *of a well-extended MSC $M = (E, \preceq, \lambda, (g_e)_{e \in E})$ if for all events $e \in E$, and for all $\alpha \in \Delta$ s.t $\alpha^M(e)$ is defined, we have $|t(\alpha^M(e)) - t(e)| \in g_e(\alpha)$.*

In the above definition, for an event $e \in E$ belonging to the domain of α, we can either have $e \prec \alpha^M(e)$ or $\alpha^M(e) \prec e$ and this would determine if the value $t(\alpha^M(e)) - t(e)$ is positive or negative. This is the reason we use the absolute value since we want only the absolute difference in the time stamps of the related events, regardless of which occurs first.

Example 8 *Consider the well-extended MSC M of Example 2 and Figure 6. Then any T-MSC over the same set of events, ordering and labelling whose time-stamps (i.e, timing function t) are such that $t(e') - t(e) \in [0, 2]$ will be a realization of M.*

Finally we note here that it is possible to extend this notion of well-extended MSCs to MSCs which have intervals between their events as timing constraints. We will look at the advantages and disadvantages of this approach in Section 8.

4.5 Proper interval alphabet and extended-MSCs

In this sub-section we introduce the notion of "proper" interval sets and alphabets and describe some properties which make them attractive. In the subsequent sections we will see the important role played by this simple notion. We say a finite set of intervals $\mathcal{S} \subseteq \mathcal{I}$ is *proper* if it forms a finite partition of $\mathbb{R}^{\geq 0} \cup \{\perp\}$. (This implies that $\mathcal{S}$ must contain $[\perp, \perp]$). Thus, if $\mathcal{S}$ is a proper interval set, then for each $t \in \mathbb{R}^{\geq 0}$, there exists a unique $I \in \mathcal{S}$ such that $t \in I$, and the intervals of $\mathcal{S}$ are either identical or disjoint from each other. An interval alphabet Γ based on Σ is said to be *proper* if for each $\alpha \in \Delta$, the set $\Gamma_\alpha = \{I \mid \exists (a, g) \in \Gamma \text{ with } g(\alpha) = I \} \cup \{[\perp, \perp]\}$ is a proper interval set. We say that an interval set *refines* another interval set if every interval of the latter is the union of some collection of intervals of the former. Finally an interval alphabet Γ *refines* another interval alphabet Γ' (both based on Σ) if Γ_α refines Γ'_α for each $\alpha \in \Delta$.

We now define for an interval set S the notion of a canonical *proper interval set of S* that refines S and is denoted $prop(S)$. Let $R = (r_0, r_1, ...r_n)$ be the sequence of bounds that appear in $\mathcal{S}$, arranged in increasing order $r_0 < r_1 < ... < r_n$ and which are different from 0,∞,$\perp$. If R is empty , we define $prop(\mathcal{S}) = \{[\perp, \perp], [0, \infty)\}$. Else, we define $prop(\mathcal{S}) = \{[\perp, \perp]\} \cup \{[0, r_0)\} \cup \{[r_i, r_i], (r_i, r_{i+1}) \mid 0 \leq i < n\} \cup \{[r_n, r_n], (r_n, \infty)\}$. Then clearly every former interval is a finite union of a set of these intervals. Thus every non-empty finite interval set $\mathcal{S}$ induces in a canonical way a proper interval set $prop(S)$ which refines it.

We now want to lift the above notion to interval alphabets. For this we are given Γ we have interval sets Γ_α for all $\alpha \in \Delta$. From this we get the proper interval sets $prop(\Gamma_\alpha)$ which cover $\Gamma_\alpha \forall \alpha \in \Delta$. Now we define a canonical *proper interval alphabet of* Γ, $prop(\Gamma) = \{(a, g) \mid a \in \Sigma$ and $\forall \alpha \in \Delta, g(\alpha) \in prop(\Gamma_\alpha)\}$. This refines Γ by definition. Thus we see that for every interval alphabet Γ there exists a proper interval alphabet which refines Γ.

Now with the above definitions we can conclude an important property of proper interval alphabets.

Lemma 9 *Let Γ be a proper interval alphabet based on Ag. Then for each TMSC T over Ag, there exists a unique well-extended MSC M (with respect to Γ) such that T is a realization of M. We denote this unique well-extended MSC as M_Γ^T.*

Proof. Let $T = (E, \preceq, \lambda, t)$. Now Γ is a proper interval alphabet which implies that for each $\alpha \in \Delta$, Γ_α is a proper interval set. This means that for each $\alpha \in \Delta$ and for each event $e \in E$, there exists a unique interval $g_e(\alpha) \in \Gamma_\alpha$ such that $|t(\alpha^M(e) - t(e)| \in g_e(\alpha)$. ($\alpha^M(e)$ is not defined means that $g_e(\alpha)$ is assigned $[\bot, \bot]$). This follows because from definition of a proper interval set any real number occurs in exactly one constituent interval. With this we look at the $\mathcal{I}^\Delta$-extended MSC M_Γ^T defined by $(E, \preceq, \lambda')$ where $\forall e \in E, \lambda'(e) = (\lambda(e), (g_e))$. Further, it is easy to observe that M_Γ^T is a well-extended MSC. Finally with this definition it follows that T is a realization of M_Γ^T. The uniqueness follows because of the properness of the interval alphabet. □

Corollary 10 *If T is $\exists^u$-B-bounded, then M_Γ^T is $\exists$-B-bounded.*

Proof. This follows because the definition of T and M_Γ^T have the same underlying MSC, i.e they have the same tuple $(E, \prec, \lambda)$. □

5 The approach via logic

Logic is a classical formalism to describe properties of various structures including words, trees, graphs etc. In this case we use it as a means to describe sets of MSCs. In particular we look at Monadic second order logic as way to describe these objects. First we look at a slightly more general case of the logic over any labelled partial order and then we restrict to *Act*-labelled as a special case. The syntax and semantics are as follows.

5.1 Monadic Second Order Logic

5.1.1 Syntax

Let Σ be an alphabet. We denote individual variables by $x, y, ...$ and set variables by $X, Y, ...$ These variables range over events or sets of events of an MSC . We also use the predicates $P_a(x)$, where x is a variable, $a \in \Sigma$. Then the set MSO(Σ) of all Monadic second order logic formulae over Σ is generated inductively using the following grammar:

$$\varphi \ ::= \ P_a(x) \mid x \in X \mid x \leq y \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x\varphi \mid \exists X\varphi$$

5.1.2 Semantics in terms of Σ-labelled partial orders

A formula φ from the above logic is interpreted on a Σ-labelled partial order $L = (E, \preceq, \lambda)$. For this, we have an interpretation $\mathbb{I}$ which is a map assigning individual variables to elements of set E and sets of variables to subsets of E. We now define the satisfaction relation $L, \mathbb{I} \models \varphi$ inductively as follows:
$L, \mathbb{I} \models P_a(x)$ if $\lambda(\mathbb{I}(x)) = a$. for $a \in \Sigma$.
$L, \mathbb{I} \models x \in X$ if $\mathbb{I}(x) \in \mathbb{I}(X)$
$L, \mathbb{I} \models x \leq y$ if $\mathbb{I}(x) \preceq \mathbb{I}(y)$
$L, \mathbb{I} \models \neg\varphi$ if $L, \mathbb{I} \not\models \varphi$
$L, \mathbb{I} \models \varphi_1 \vee \varphi_2$ if $L, \mathbb{I} \models \varphi_1$ or $L, \mathbb{I} \models \varphi_2$
$L, \mathbb{I} \models \exists x\varphi$ if $\exists e \in E$ such that $L, \mathbb{I}[x \mapsto e] \models \varphi$
$L, \mathbb{I} \models \exists X\varphi$ if $\exists E' \subseteq E$ such that $L, \mathbb{I}[X \mapsto E'] \models \varphi$.

For sentences φ in this logic (i.e, those that have no free variables), we write $L \models \varphi$ instead of $L, \mathbb{I} \models \varphi$. Note that here and henceforth $\mathbb{I}[x \to e]$ denotes the interpretation $\mathbb{I}'$ s.t $\mathbb{I}'(x) = e$ and for all var $y \neq x$, $\mathbb{I}'(y) = \mathbb{I}(y)$. Similarly $\mathbb{I}[X \to E']$ is the interpretation which maps any element of X to an element of E' but coincides with $\mathbb{I}$ for all other variables outside set X.

Now if we choose Σ to be the set of actions Act, then we get the semantics in terms of MSCs over Ag. Further, if we choose $\Sigma = Act \times \Sigma'$, we get the semantics in terms of Σ'-extended MSCs over Ag. If we choose Σ' to be $\mathcal{I}^{\Delta}$, we get an interval alphabet $\Gamma = Act \times \mathcal{I}^{\Delta}$. Thus we have MSO($\Gamma$) formulae whose semantics are in terms of $\mathcal{I}^{\Delta}$-extended MSCs over Ag. Thus for a sentence φ in MSO(Γ), we have $\mathcal{L}_{ext}(\varphi) = \{M \in \mathbb{MSC}(\Gamma) \mid M \models \varphi\}$.

5.2 Timed Monadic Second Order logic

5.2.1 Syntax

In addition to the predicates and variables used in the untimed case, here we use the predicates $\alpha(x) \in I$, where x is a variable, $I \in \mathcal{I}, \alpha \in \Delta$. In using this predicate we intuitively mean that the time difference between the event associated with x and $\alpha(x)$ under some interpretation is contained

in the interval I. Then the formal syntax is given by:

$$\varphi \ ::= \ P_a(x) \mid x \in X \mid x \leq y \mid \alpha(x) \in I \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x\varphi \mid \exists X\varphi$$

We call the logic so defined TMSO($Act, \mathcal{I}, \Delta$). However when it is clear from the context we will just use TMSO for referring to formulas from this logic.

5.2.2 Semantics in terms of T-MSCs

A formula φ from this logic over Act is interpreted on a T-MSC $T = (E, \preceq, \lambda, t)$ over Ag.
So, as before given a T-MSC T, we have an interpretation $\mathbb{I}$. Then the satisfaction relation $T, \mathbb{I} \models \varphi$ is defined inductively by:
$T, \mathbb{I} \models P_a(x)$ if $\lambda(\mathbb{I}(x)) = a$,
$T, \mathbb{I} \models x \in X$ if $\mathbb{I}(x) \in \mathbb{I}(X)$
$T, \mathbb{I} \models x \leq y$ if $\mathbb{I}(x) \preceq \mathbb{I}(y)$
$T, \mathbb{I} \models \alpha(x) \in I$ if $\mathbb{I}(x)$ is in the domain of α^T and $|t(\mathbb{I}(x)) - t(\alpha^T(\mathbb{I}(x)))| \in I$,
or $\mathbb{I}(x)$ is not in domain of α^T and $I = [\perp, \perp]$.
$T, \mathbb{I} \models \neg\varphi$ if $T, \mathbb{I} \not\models \varphi$
$T, \mathbb{I} \models \varphi_1 \vee \varphi_2$ if $T, \mathbb{I} \models \varphi_1$ or $T, \mathbb{I} \models \varphi_2$
$T, \mathbb{I} \models \exists x\varphi$ if $\exists e \in E$ such that $T, \mathbb{I}[x \mapsto e] \models \varphi$
$T, \mathbb{I} \models \exists X\varphi$ if $\exists E' \subseteq E$ such that $T, \mathbb{I}[X \mapsto E'] \models \varphi$.

We define for sentences φ in this logic, $\mathcal{L}_{time}(\varphi) = \{T \in \mathbb{TMSC} \mid T \models \varphi\}$.

5.3 TMSO to MSO(Γ) Translation

Given a TMSO formula φ, we denote by $voc_\alpha(\varphi)$, the set of intervals I, for which there exists a sub-formula of φ of the form $\alpha(x) \in I$. Let $\Gamma(\varphi) = \{(a, g) \mid a \in Act, \forall \alpha \in \Delta, g(\alpha) \in prop(voc_\alpha(\varphi))\}$. Thus $\Gamma(\varphi)$ is a proper interval alphabet over Act such that for each $\alpha \in \Delta, \Gamma(\varphi)_\alpha$ refines $voc_\alpha(\varphi)$

Now we obtain a MSO($\Gamma(\varphi)$) formula, φ^{prop} by replacing each sub-formula of the form:

- $P_a(x)$ by the formula $\bigvee_{(b,g)\in\Gamma(\varphi), b=a} P_{(b,g)}(x)$,
- $\alpha(x) \in I$ by the formula $\bigvee_{(b,g)\in\Gamma(\varphi), g(\alpha)\subseteq I} P_{(b,g)}(x)$,

Recall that given a T-MSC T and an interval alphabet Γ, by Lemma 9, there exists a uniquely defined well-extended MSC M_Γ^T. Now fixing the interval alphabet to be $\Gamma(\varphi)$ we have the following lemma.

Lemma 11 *For any T-MSC T over Ag and a TMSO φ over Act, $T \models \varphi$ iff $M^T_{\Gamma(\varphi)} \models \varphi^{prop}$.*

Proof. Let $T = (E, \preceq, \lambda, t)$ be a T-MSC. From TMSO φ we get an interval alphabet $\Gamma(\varphi)$. Then by Lemma 9 (and its proof) we have the unique well-extended MSC $M^T_{\Gamma(\varphi)} = (E, \preceq, \lambda')$, where $\lambda'(e) = (\lambda(e), g_e)$ s.t for all $\alpha, |t(\alpha^{(M^T_{\Gamma(\varphi)})}(e)) - t(e)| \in g_e(\alpha)$ if $\alpha^{(M^T_{\Gamma(\varphi)})}(e)$ is defined and $g_e(\alpha) = [\perp, \perp]$ otherwise. Note that α^T and $\alpha^{(M^T_{\Gamma(\varphi)})}$ are identical and we will henceforth use only the former notation. Now, we show the lemma for any interpretation $\mathbb{I}$. Proof is by induction on structure of φ. The only interesting cases are case(i) and case(iv). The others are routine deductions sketched here for the sake of completeness.

- *Case(i)* : $\varphi = P_a(x)$
  $T, \mathbb{I} \models \varphi \iff \lambda(\mathbb{I}(x)) = a$
  $\iff M^T_{\Gamma(\varphi)}, \mathbb{I} \models P_{(a,g)}(x)$ for some g such that $(a, g) \in \Gamma(\varphi)$.
  $\iff M^T_{\Gamma(\varphi)}, \mathbb{I} \models \bigvee_{(b,g)\in\Gamma(\varphi), b=a} P_{(b,g)}(x)$.
  $\iff M^T_{\Gamma(\varphi)}, \mathbb{I} \models \varphi^{prop}$.

- *Case(ii)* : $x \in X$. This is trivial by definition since the interpretations assign the same events to variables in both T and $M^T_{\Gamma(\varphi)}$.
  $T, \mathbb{I} \models x \in X \iff \mathbb{I}(x) \in \mathbb{I}(X) \iff M^T_{\Gamma(\varphi)}, \mathbb{I} \models x \in X$.

- *Case(iii)*: $x \leq y$. Trivial as in case (ii).

- *Case(iv)* : $\varphi = \alpha(x) \in I$
  $T, \mathbb{I} \models \varphi \implies T, \mathbb{I} \models \alpha(x) \in I \implies$ Either $\alpha^T(\mathbb{I}(x))$ is defined and $|t(\mathbb{I}(x)) - t(\alpha^T(\mathbb{I}(x)))| \in I$ or $\alpha^T(\mathbb{I}(x))$ is undefined and $I = [\perp, \perp]$. Further we already have that in $M^T_{\Gamma(\varphi)}$, $|t(\mathbb{I}(x)) - t(\alpha^{(M^T_{\Gamma(\varphi)})}(\mathbb{I}(x)))| \in g_{\mathbb{I}(x)}(\alpha)$ and that $g_{\mathbb{I}(x)}(\alpha) = [\perp, \perp]$ iff α^T is not defined on $\mathbb{I}(x)$.
  Now using the fact that $\Gamma_\alpha(\varphi)$ is proper and refines voc_a, it must be the case that either $\mathbb{I}(x)$ is in the domain of α^T and $g_{\mathbb{I}(x)}(\alpha) \subseteq I$ or it is not in the domain, in which case, $g_{\mathbb{I}(x)}(\alpha) = I = [\perp, \perp]$. In both cases we can write,
  $M^T_{\Gamma(\varphi)}, \mathbb{I} \models P_{(\lambda(\mathbb{I}(x)), g_{\mathbb{I}(x)})}(x)$ s.t $g_{\mathbb{I}(x)}(\alpha) \subseteq I$.
  $\implies M^T_{\Gamma(\varphi)}, \mathbb{I} \models \bigvee_{(b,g)\in\Gamma(\varphi), g(\alpha)\subseteq I} P_{(b,g)}(x)$.
  $\implies M^T_{\Gamma(\varphi)}, \mathbb{I} \models \varphi^{prop}$

  Conversely, let $M^T_{\Gamma(\varphi)}, \mathbb{I} \models \varphi^{prop}$
  $\implies M^T_{\Gamma(\varphi)}, \mathbb{I} \models \bigvee_{(b,g)\in\Gamma(\varphi), g(\alpha)\subseteq I} P_{(b,g)}(x)$.

$\implies \lambda'(\mathbb{I}(x)) = (b, g_{\mathbb{I}(x)})$ for some $(b, g_{\mathbb{I}(x)}) \in \Gamma(\varphi)$ such that $g_{\mathbb{I}(x)}(\alpha) \subseteq I$. But T is a realization of $M^T_{\Gamma(\varphi)}$ implies that either $\alpha^T(\mathbb{I}(x))$ is defined and $|t(\alpha^T(\mathbb{I}(x)) - t(\mathbb{I}(x))| \in I$ or it is not defined, in which case $g_{\mathbb{I}(x)}(\alpha) = [\bot, \bot]$ and thus $I = [\bot, \bot]$(else we will not have $g_{\mathbb{I}(x)}(\alpha) \subseteq I$).
Thus we have either $\mathbb{I}(x)$ is in the domain of α^T and $|t(\alpha^T(\mathbb{I}(x)) - t(\mathbb{I}(x))| \in I$ or it is not in the domain and $I = [\bot, \bot]$ and therefore we conclude that $T, \mathbb{I} \models \alpha(x) \in I$.

- *Case(v)*: negation, $\neg\varphi$
  $T, \mathbb{I} \models \neg\varphi \iff T, \mathbb{I} \not\models \varphi$. But φ is a formula strictly smaller (with less structure) than $\neg\varphi$, so we can apply the induction hypothesis.This means that $T, \mathbb{I} \models \varphi \iff M^T_{\Gamma(\varphi)}, \mathbb{I} \models \varphi$ which implies $T, \mathbb{I} \not\models \varphi \iff M^T_{\Gamma(\varphi)}, \mathbb{I} \not\models \varphi$. Thus, $T, \mathbb{I} \models \neg\varphi \iff T, \mathbb{I} \not\models \varphi \iff M^T_{\Gamma(\varphi)}, \mathbb{I} \not\models \varphi \iff M^T_{\Gamma(\varphi)}, \mathbb{I} \models \neg\varphi$.

- *Case(vi)* : or, $\varphi = \varphi_1 \vee \varphi_2$
  $T, \mathbb{I} \models \varphi_1 \vee \varphi_2 \iff T, \mathbb{I} \models \varphi_1$ or $T, \mathbb{I} \models \varphi_2 \iff M^T_{\Gamma(\varphi)}, \mathbb{I} \models \varphi_1$ or $M^T_{\Gamma(\varphi)}, \mathbb{I} \models \varphi_2$ (by induction hyp. since both φ_1 and φ_2 are smaller than φ) $\iff M^T_{\Gamma(\varphi)}, \mathbb{I} \models \varphi_1 \vee \varphi_2$.

- *Case(vii)* : $\varphi = \exists x \varphi_1$
  $T, \mathbb{I} \models \varphi \iff \exists e \in E$ s.t $T, \mathbb{I}[x \to e] \models \varphi_1$. But φ_1 is strictly smaller than φ and interpretation assigns same values to variables in both cases, so by ind hyp. we have $T, \mathbb{I}[x \to e] \models \varphi_1 \iff M^T_{\Gamma(\varphi)}, \mathbb{I}[x \to e] \models \varphi_1$. So, $\exists e \in E$ s.t $T, \mathbb{I}[x \to e] \models \varphi_1 \iff \exists e \in E$ (in fact the same e as referred to by T) s.t $M^T_{\Gamma(\varphi)}, \mathbb{I}[x \to e] \models \varphi_1 \iff M^T_{\Gamma(\varphi)}, \mathbb{I} \models \exists x \varphi_1$.

- *Case(viii)* : $\varphi = \exists X \varphi_1$.
  The proof in this case is exactly the same as in case(vii) above with events replaced by sets of events. □

5.4 MSO to TMSO Translation

Given alphabet *Act* and timing alphabetΔ, we now show how to associate with a given MSO($Act \times \mathcal{I}^\Delta$) formula φ a TMSO formula φ^t over *Act* which has the same set of timed realizations. We get TMSO formula φ^t by replacing

atomic sub-formulas of φ of the form $P_{(a,g)}(x)$ (where $a \in Act, g \in \mathcal{I}^{\Delta}$) by:

$$P_a(x) \ \wedge \bigwedge_{\alpha \in \Delta} \alpha(x) \in g(\alpha)$$

Then we have the following lemma,

Lemma 12 *Let Γ be a proper interval alphabet based on Act and let φ be sentence in MSO(Γ). Then for a given T-MSC T, $T \models \varphi^t$ iff $M^T_{\Gamma(\varphi)} \models \varphi$.*

Proof. The proof is by straightforward structural induction along the lines of Lemma 11. Hence we will prove the statement for the only interesting case of $\varphi = P_{(a,g)}(x)$. Let $T = (E, \preceq, \lambda, t)$ be a T-MSC over Ag, and $M^T_\Gamma = (E, \preceq, \lambda, (g'_e)_{e \in E})$ be obtained as in Lemma 9 from T and Γ.
We will prove that for any interpretation $\mathbb{I}$, $T, \mathbb{I} \models \varphi^t$ iff $M^T_\Gamma, \mathbb{I} \models \varphi$
$(\implies)$
$T, \mathbb{I} \models \varphi^t \implies T, \mathbb{I} \models P_a(x) \wedge \bigwedge_{\alpha \in \Delta} \alpha(x) \in g(\alpha)$
$\implies T, \mathbb{I} \models P_a(x)$ and $T, \mathbb{I} \models \bigwedge_{\alpha \in \Delta} \alpha(x) \in g(\alpha)$
$\implies \lambda(\mathbb{I}(x)) = a$ and for all $\alpha \in \Delta$, either $\mathbb{I}(x)$ is in the domain of α^T and $|t(\mathbb{I}(x)) - t(\alpha^T(\mathbb{I}(x)))| \in g(\alpha)$, or $\mathbb{I}(x)$ is not in the domain of α^T and $g(\alpha) = [\perp, \perp]$. By properness of the interval alphabet Γ we conclude that $g = g'_{\mathbb{I}(x)}$ and so $M^T_\Gamma, \mathbb{I} \models P_{(a,g)}(x)$.
$(\impliedby)$
Conversely,
$M^T_\Gamma, \mathbb{I} \models P_{(a,g)}(x)$ and the fact that T is a realization of M^T_Γ together imply that $\lambda(\mathbb{I}(x)) = a$ and for all $\alpha \in \Delta$, either $\mathbb{I}(x)$ is in the domain of α^T and $|t(\mathbb{I}(x)) - t(\alpha^T(\mathbb{I}(x)))| \in g_{\mathbb{I}(x)}(\alpha)$, or $\mathbb{I}(x)$ is not in the domain of α^T and $g_{\mathbb{I}(x)}(\alpha) = [\perp, \perp]$.
$\implies T, \mathbb{I} \models P_a(x)$ and $T, \mathbb{I} \models \bigwedge_{\alpha \in \Delta} \alpha(x) \in g(\alpha) \implies T, \mathbb{I} \models P_a(x) \wedge \bigwedge_{\alpha \in \Delta} \alpha(x) \in g(\alpha)$. Thus $T, \mathbb{I} \models \varphi^t$ and we are done. □

6 The automaton model

The most natural formalism to describe (asynchronous) communication protocols are communicating finite state machines (CFM) [BZ83, GKM06] which can be seen as a set of finite state machines with message passing channels between any pair of them. For our purposes we need a slight generalization of a CFM which is over an extended alphabet as defined below. Also though we define them over the set of processes Ag, as in the earlier case we will not write this explicitly each time.

Definition 13 (ext-CFM) *A* Σ-extended Communicating Finite-state Machine *is a tuple $\mathcal{A} = (C, (\mathcal{A}_p)_{p \in Ag}, F)$, where*

- *C is a finite set of message contents or control messages.*
- *Each $\mathcal{A}_p = (Q_p, \rightarrow_p, i_p)$ is a finite transition system over the alphabet $(Act \times \Sigma) \times C$ for any process $p \in Ag$ (i.e, $\rightarrow_p$ is a finite subset of $Q_p \times (Act \times \Sigma \times C) \times Q_p$) with initial state $i_p \in Q_p$.*
- *$F \subseteq \prod_{p \in Ag} Q_p$ is a set of global final states.*

To define the semantics of a Σ-extended CFM $\mathcal{A}$ we use Σ-extended MSCs to represent successful runs. The partial order run of an extended CFM is defined as an extended MSC with the events being labelled consistently by local states of the machine. Formally let $M = (E, \preceq, \lambda')$ be a Σ-extended MSC over Ag and $\rho : E \rightarrow \bigcup_{p \in Ag} Q_p$ be a map labeling each event of process p with a local state from Q_p. More precisely, $\rho(e) \in Q_{(Ag(e))}$ for all $e \in E$. Now we define $\rho^- : E \rightarrow \bigcup_{p \in Ag} Q_p$ as follows: For e in E, if there is an e' in E such that $Ag(e) = Ag(e')$ and $e' \prec_p e$, then we set $\rho^-(e) = \rho(e')$. Otherwise (e is the minimal event in the p^{th} process), we set $\rho^-(e) = i_p$. Then ρ is said to be a *run* of $\mathcal{A}$ on M if for $e \in S$ (set of send events), $e' \in R$ (the set of receive events) and $a, b \in \Sigma$, with $\lambda'(e) = (p!q, a)$, $Msg(e) = e'$ and $\lambda'(e') = (q?p, b)$, there is a control message $c \in C$ s.t

$$\rho^-(e) \xrightarrow{(p!q,a),c}_{Ag(e)} \rho(e) \quad \text{and} \quad \rho^-(e') \xrightarrow{(q?p,b),c}_{Ag(e')} \rho(e')$$

Let ρ be a run of $\mathcal{A}$ on M. Define $s_p = \rho(e_p)$, where e_p is the maximal event in $(E_p, \prec_p)$, if E_p is non-empty. Else set $s_p = i_p$. Then run ρ is *successful* if the tuple $(s_p)_{p \in Ag}$ belongs to the set of global final states F. A Σ-extended MSC is *accepted* by a Σ-extended CFM $\mathcal{A}$ if it admits a successful run. The set of all such MSCs in denoted $\mathcal{L}_{ext}(\mathcal{A})$.

The reason for introducing these message contents is the same as for CFMs in [GKM06], which is that without them we do not achieve the full expressive power of MSO logic or in our case, the TMSO logic.

6.1 The timed model

Now to introduce the timed model we attach finitely many real-valued variables called *clocks* to these machines. This means that each constituent finite state machine is a timed automaton that can accept timed words with the only difference that the clocks (or timing constraints) may refer to events in the other constituent CFMs and thus are in some sense "global".

In our model we will use one clock for each symbol in the timing alphabet. And since we will know by context whether we refer to a clock or a symbol of the timing alphabet and since they are in a one-to-one correspondence, we use the same set of symbols to denote the set of clocks, $\alpha \in \Delta$. Formally, this automaton model is nothing but an extended CFM whose extended alphabet is a set of clocks each of which maps to an interval, in other words,

Definition 14 (EC-CFM) *An* Event Clock Communicating Finite-state Machine *is a $\mathcal{I}^{\Delta}$-extended CFM. That is, $\mathcal{A} = (C, (\mathcal{A}_p)_{p \in Ag}, F)$ where, C is a finite set of message contents, F is a set of global final states and each $\mathcal{A}_p$ is an finite labelled transition system $(Q_p, \rightarrow_p, i_p)$ over $Act_p \times \mathcal{I}^{\Delta} \times C$. The transition relation $\rightarrow_p$ is defined to be a finite subset of the set $Q_p \times Act_p \times \mathcal{I}^{\Delta} \times C \times Q_p$.*

The semantics of these EC-CFMs in terms of $\mathcal{I}^{\Delta}$-extended MSCs is given by Definition 13.

6.2 Semantics in terms of T-MSCs

On the other hand EC-CFMs can be realized in terms of T-MSCs as well. This alternative characterization is more useful while looking at the EC-CFMs as Communicating finite state machines with clocks attached. For the definition of semantics of an EC-CFM, we can use T-MSCs to represent successful runs. This semantics is used when we want to interpret each $\mathcal{A}_p$ as a finite automaton with clocks (local or global). Thus here the clock constraints are interpreted with respect to clock-valuation functions. We make a small note that the event recording clocks as defined in [AHP96] correspond to the symbol $Prev_a$ as defined in example 1. The issue of implementability of these clocked automata will be looked at in section 9.

The semantics is as follows. Let $T = (E, \preceq, \lambda, t)$ be a T-MSC over Ag. We define ρ and ρ^- exactly as in the case of CFMs over MSCs. (This means that we let $\rho : E \rightarrow \bigcup_{p \in Ag} Q_p$ be a map labeling each event of the T-MSC with a state of a constituent automaton of the EC-CFM. More precisely, $\rho(e) \in Q_{(Ag(e))}$ for all $e \in E$. Recall that we define $\rho^- : E \rightarrow \bigcup_{p \in Ag} Q_p$ as follows: For e in E, if there is an e' in E such that $Ag(e) = Ag(e')$ and $e' \prec_p e$, then we set $\rho^-(e) = \rho(e')$. Otherwise we set $\rho^-(e) = i_p$.)

Then ρ is said to be a *run* of $\mathcal{A}$ on T-MSC T if for $e \in S$ and $e' \in R$ with $\lambda(e) = p!q$ and $Msg(e) = e'$, there are $g, g' : \Delta \rightarrow \mathcal{I}$ and a control message $c \in C$ s.t

$$\rho^-(e) \xrightarrow{p!q,g,c}_{Ag(e)} \rho(e) \quad \text{and} \quad \rho^-(e') \xrightarrow{q?p,g',c}_{Ag(e')} \rho(e')$$

with $|t(e) - t(\alpha^T(e))| \in g(\alpha)$, for all $\alpha \in \Delta$ s.t e is in the domain of α^T, and similarly for e', $|t(e') - t(\alpha^T(e'))| \in g'(\alpha)$, for all $\alpha \in \Delta$ s.t e' is in the domain of α^T. Additionally g (or g') maps to $[\perp, \perp]$ if and only if $\alpha^T(e)$ (or $\alpha^T(e')$) is undefined.

The definition of successful runs and the acceptance criterion are as before.
That is, if ρ is a run of $\mathcal{A}$ on T, define $s_p = \rho(e_p)$, where e_p is the maximal event in $(E_p, \prec_p)$, if E_p is non-empty. Else set $s_p = i_p$. Then run ρ is *successful* if the tuple $(s_p)_{p \in Ag}$ belongs to the set of global final states F. A T-MSC is *accepted* by a EC-CFM $\mathcal{A}$ if it admits a successful run. We denote by $\mathcal{L}_{time}(\mathcal{A})$, the set of T-MSCs that are accepted by $\mathcal{A}$.

6.3 Translation to automata over a proper interval alphabet

Now given an EC-CFM we have two different semantics, one in terms of timed MSCs and the other in terms of $\mathcal{I}^{\Delta}$-extended MSCs. As in the case of logic, we will translate any EC-CFM into another which has a proper interval alphabet and behaves correspondingly (as characterized by the lemma below).

To do this we first observe that given an EC-CFM $\mathcal{A}$, since it is defined as a finite system with a finite transition relation, we have for each clock α a finite set of intervals occurring as guards for it. Let us denote this set $voc_\alpha(\mathcal{A})$ for each α. Then we define $\Gamma(\mathcal{A}) = \{(a, g) \mid a \in Act, \forall \alpha \in \Delta, g(\alpha) \in prop(voc_\alpha(\mathcal{A}))\}$. Thus $\Gamma(\mathcal{A})$ is a proper interval alphabet over Act such that $\Gamma(\mathcal{A})_\alpha$ refines $voc_\alpha(\mathcal{A})$ for each α.

Now suppose $\mathcal{A} = (C, (\mathcal{A}_p)_{p\in Ag}, F)$ where $\mathcal{A}_p = (Q_p, \rightarrow_p, i_p)$ over $Act_p \times \mathcal{I}^{\Delta} \times C$. Then we define $\mathcal{A}^{prop} = (C, (\mathcal{A}'_p)_{p\in Ag}, F)$ where $\mathcal{A}'_p = (Q_p, \rightarrow'_p, i_p)$ over $Act_p \times \mathcal{I}^{\Delta} \times C$. Thus the only difference between the automata are the transition relations. The transition relation of $\mathcal{A}^{prop}$, $\rightarrow'_p$ is defined as follows,
$(q, a, g', c, q') \in \rightarrow'_p$ if $g'(\alpha) \in prop(voc_\alpha(\mathcal{A}))$ for all $\alpha \in \Delta$ and there exists a transition $(q, a, g, c, q') \in \rightarrow_p$ such that $g'(\alpha) \subseteq g(\alpha)$ for all $\alpha \in \Delta$.

Thus for each transition of $\mathcal{A}$, $\mathcal{A}^{prop}$ has a set of transitions that are labelled by intervals from a proper interval set and are contained in the corresponding interval of the transition of $\mathcal{A}$. Then we have the following lemma,

Lemma 15 *Given a T-MSC T and EC-CFM $\mathcal{A}$, $T \in \mathcal{L}_{time}(\mathcal{A})$ iff $M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A}^{prop})$.*

Proof. Let $T = (E, \preceq, \lambda, t)$ and $\mathcal{A} = (C, (\mathcal{A}_p)_{p\in Ag}, F)$ with $\mathcal{A}_p = (Q_p, \rightarrow_p, i_p)$.
($\Longrightarrow$) $T \in \mathcal{L}_{time}(\mathcal{A})$ implies that there is a run ρ of $\mathcal{A}$ on T such that for any $e \in E$ there is a $g : \Delta \rightarrow \mathcal{I}$ and a control message $c \in C$ s.t,

$$\rho^-(e) \xrightarrow{\lambda(e),g,c}_{Ag(e)} \rho(e)$$

with $|t(e) - t(\alpha^T(e))| \in g(\alpha)$ for all $\alpha \in \Delta$ s.t e is in the domain of α^T and $g(\alpha) = [\perp, \perp]$ otherwise.
Now whenever $\alpha^T(e)$ is defined, each $g(\alpha) \in voc_\alpha(\mathcal{A})$. Then by the definition of properness we have that $|t(e) - t(\alpha^T(e))| \in I_\alpha$ for a unique $I_\alpha \in prop(voc_\alpha(\mathcal{A}))$ such that $I_\alpha \subseteq g(\alpha)$. Now we define $g'_e : \Delta \rightarrow \mathcal{I}$ with $g'_e(\alpha) = I_\alpha$ if e is the domain of α and $g'_e(\alpha) = [\perp, \perp]$ otherwise.

Then ρ is a run of $\mathcal{A}^{prop}$ on ext-MSC $M = (E, \preceq, \lambda, (g'_e)_{e\in E})$ since,

$$\rho^-(e) \xrightarrow{\lambda(e),g'_e,c}_{Ag(e)} \rho(e)$$

This follows from the definition of transitions of $\mathcal{A}^{prop}$. Now we observe that M is over the proper interval alphabet $\Gamma(\mathcal{A})$ and that M is well-extended. Also T is a realization of M, since for all $e \in E$, and $\alpha \in \Delta$ such that $\alpha^M(e)$ is defined, we have $|t(\alpha^M(e)) - t(e)| = |t(\alpha^T(e)) - t(e)| \in I_\alpha = g'_e(\alpha)$. Now from Lemma 9, we have uniqueness and we conclude that $M = M^T_{\Gamma(\mathcal{A})}$.

Thus $\mathcal{A}$ has a run on T implies that $\mathcal{A}^{prop}$ has a run on $M^T_{\Gamma(\mathcal{A})}$ and so $T \in \mathcal{L}_{time}(\mathcal{A}) \implies M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A}^{prop})$.

($\Longleftarrow$) Conversely, if we have a run ρ of $\mathcal{A}^{prop} = (C, (\mathcal{A}'_p)_{p \in Ag}, F)$ on $M^T_{\Gamma(\mathcal{A})} = (E, \preceq, \lambda, (g'_e)_{e \in E})$. This means that for any $e \in E$ there is a control message $c \in C$ s.t,

$$\rho^-(e) \xrightarrow{\lambda(e), g'_e, c}_{Ag(e)} \rho(e)$$

But by definition of the $\mathcal{A}^{prop}$, in $\mathcal{A}$ there exists g such that there is a transition,

$$\rho^-(e) \xrightarrow{\lambda(e), g, c}_{Ag(e)} \rho(e)$$

with $g'_e(\alpha) \subseteq g(\alpha)$ for all $\alpha \in \Delta$.
Also T is a realization of $M^T_{\Gamma(\mathcal{A})}$ implies for all $e \in E$, and $\alpha \in \Delta$ such that $\alpha^T(e)$ is defined, we have $|t(\alpha^T(e)) - t(e)| \in g'_e(\alpha) \subseteq g(\alpha)$. And for e not in the domain of α^T we have $g'_e(\alpha) = [\perp, \perp]$ which implies that $g(\alpha) = [\perp, \perp]$. This implies that ρ is a run on T of $\mathcal{A}$.

Thus given a run of $\mathcal{A}^{prop}$ on $M^T_{\Gamma(\mathcal{A})}$ we have shown the existence of a run of $\mathcal{A}$ on T. So we conclude that $M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A}^{prop}) \implies T \in \mathcal{L}_{time}(\mathcal{A})$

□

Corollary 16 *If $\mathcal{A}$ is an EC-CFM such that $voc_\alpha(\mathcal{A})$ is a proper interval set for all $\alpha \in \Delta$, then $T \in \mathcal{L}_{time}(\mathcal{A})$ iff $M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A})$.*

Proof. This follows trivially from the Lemma 15 above since if $voc_\alpha(\mathcal{A})$ is proper then $prop(voc_\alpha(\mathcal{A})) = voc_\alpha(\mathcal{A})$ for all α and therefore we have $\mathcal{A} = \mathcal{A}^{prop}$. □

7 Equivalence of automata and logic over $\exists^u$-bounded timed MSCs

In the recent paper by Genest, Kuske and Muscholl [GKM06], the equivalence between MSO formulas and CFMs over MSCs has been described in the context of (existentially) bounded channels. The approach they have used is of reinterpreting these as Mazurkiewicz traces [DR95] and applying proof techniques known for traces. Here we will state (without proof) the

main theorem of that setting. Then we will try to lift that theorem to the timed setting and in doing so use the concepts that we have defined in the previous sections. First we state a slightly generalized version of the relevant theorem whose proof follows from the original theorem and its proof,

Theorem 17 ([GKM06]) *For some fixed positive integer B and alphabet Σ,*

1. *Given a MSO($Act \times \Sigma$) formula φ such that $\mathcal{L}_{ext}(\varphi)$ is $\exists$-B-bounded we can construct a Σ-extended CFM $\mathcal{A}$ such that $\mathcal{L}_{ext}(\varphi) = \mathcal{L}_{ext}(\mathcal{A})$.*
2. *Given a Σ-extended CFM $\mathcal{A}$ such that $\mathcal{L}_{ext}(\mathcal{A})$ is $\exists$-B-bounded we can construct a MSO($Act \times \Sigma$) formula φ such that $\mathcal{L}_{ext}(\mathcal{A}) = \mathcal{L}_{ext}(\varphi)$.*

We note for this result to hold we require the presence of the finite set of message contents C in the CFM model. And since we would like to use this result to prove the following theorem/equivalence it forces our automaton model EC-CFM also to contain this set of message contents.

Now in the timed setting our approach to get such an equivalence is to convert the timed MSO and EC-CFM into MSOs and CFMs over extended alphabets interpreted over extended-MSCs. Then we can use the above result. For this conversion we use Lemma 11,12 and 15 that we have in Section 5 and 6 by extending the labels of events with intervals or sets of intervals. We move on to the formal theorem and proof below.

Theorem 18 *For some fixed positive integer B and timing alphabet Δ,*

1. *Given a TMSO formula φ such that $\mathcal{L}_{time}(\varphi)$ is $\exists^u$-B-bounded we can construct an EC-CFM $\mathcal{A}$ such that $\mathcal{L}_{time}(\varphi) = \mathcal{L}_{time}(\mathcal{A})$.*
2. *Given an EC-CFM $\mathcal{A}$ such that $\mathcal{L}_{time}(\mathcal{A})$ is $\exists^u$-B-bounded we can construct a TMSO formula φ such that $\mathcal{L}_{time}(\mathcal{A}) = \mathcal{L}_{time}(\varphi)$.*

Proof. The two parts of the theorem are converses of each other and we will show them separately as follows,

Part 1. Let φ be the given TMSO formula. Then we have $\Gamma(\varphi)$ to be a proper interval alphabet based on Act such that $\Gamma(\varphi)_\alpha$ refines $voc_\alpha(\varphi)$ for each $\alpha \in \Delta$ as defined in section 5.3. Then by the TMSO-MSO translation we get an MSO($\Gamma(\varphi)$) formula, φ^{prop} such that any T-MSC T, $T \models \varphi$ iff $M^T_{\Gamma(\varphi)} \models \varphi^{prop}$.

Now we use the fact that the set of all $\exists$-B-bounded MSCs can be described by a MSO formula. This follows from a result of Genest et al., to be precise from [GKM06, Prop. 5.14] and [GKM06, Theorem 4.1]. Let us call this formula for MSO($\Gamma(\varphi)$) to be ξ_B. Thus $\mathcal{L}_{ext}(\xi_B)$ is the set of all

$\exists$-B-bounded extended MSCs over $\Gamma(\varphi)$.

Next we look at the MSO($\Gamma(\varphi)$) formula, $(\varphi^{prop} \wedge \xi_B)$. Since $\mathcal{L}_{ext}(\varphi^{prop} \wedge \xi_B) \subseteq \mathcal{L}_{ext}(\xi_B)$ we obtain that the language $\mathcal{L}_{ext}(\varphi^{prop} \wedge \xi_B)$ is existentially-B-bounded. Now, by Theorem 17 we get an EC-CFM $\mathcal{A}$ such that $\mathcal{L}_{ext}(\mathcal{A}) = \mathcal{L}_{ext}(\varphi^{prop} \wedge \xi_B)$. Theorem 17 can be applied here because of finiteness of alphabet which is true since the intervals we look at arise from a finite set of intervals that occur in the formula, $\Gamma(\varphi)$. In addition $\Gamma(\mathcal{A}) = \Gamma(\varphi)$ since the set of intervals occurring in $\mathcal{A}$ must come from the proper interval set of φ. (and further if there are any intervals of $\Gamma(\varphi)$ missing we can add some redundant transitions to $\mathcal{A}$ which mention such intervals. This can be done since it does not change the set of ext-MSCs accepted .)

Thus we finally have the following sequence of deductions,
$T \in \mathcal{L}_{time}(\varphi) \implies T \models \varphi \implies M^T_{\Gamma(\varphi)} \models \varphi^{prop}$ by Lemma 11. Also by Corollary 10 of Lemma 9, T is $\exists^u$-B-bounded implies that $M^T_{\Gamma(\varphi)}$ is $\exists$-B-bounded. Therefore $M^T_{\Gamma(\varphi)} \models \xi_B$. Hence $M^T_{\Gamma(\varphi)} \in \mathcal{L}_{ext}(\varphi^{prop} \wedge \xi_B)$.
But from the above discussion $\mathcal{L}_{ext}(\varphi^{prop} \wedge \xi_B) = \mathcal{L}_{ext}(\mathcal{A})$. Also $\Gamma(\mathcal{A}) = \Gamma(\varphi) \implies M^T_{\Gamma(\varphi)} = M^T_{\Gamma(\mathcal{A})}$ Thus we obtain $M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A})$. Now by Corollary 16 of Lemma 15, we get $T \in \mathcal{L}_{time}(\mathcal{A})$. Therefore we have shown that $\mathcal{L}_{time}(\varphi) \subseteq \mathcal{L}_{time}(\mathcal{A})$.

Now starting from $T \in \mathcal{L}_{time}(\mathcal{A})$ we have by Lemma 15, $M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A})$. But again $\mathcal{L}_{ext}(\mathcal{A}) = \mathcal{L}_{ext}(\varphi^{prop} \wedge \xi_B)$ and $M^T_{\Gamma(\varphi)} = M^T_{\Gamma(\mathcal{A})}$. Therefore we obtain $M^T_{\Gamma(\varphi)} \in \mathcal{L}_{ext}(\varphi^{prop} \wedge \xi_B)$. Thus, $M^T_{\Gamma(\varphi)} \models \varphi^{prop}$ and by Lemma 11 conclude that $T \models \varphi$. This implies $\mathcal{L}_{time}(\mathcal{A}) \subseteq \mathcal{L}_{time}(\varphi)$.

Hence from a TMSO formula φ whose set of T-MSCs is given to be existentially-B-bounded T-MSCs we have obtained an EC-CFM which accepts exactly the same set of T-MSCs.

Part 2. For the converse we have EC-CFM $\mathcal{A}$ such that $\mathcal{L}_{time}(\mathcal{A})$ is $\exists^u$-B-bounded. Then from the set of intervals occurring in $\mathcal{A}$ we get the proper interval alphabet $\Gamma(\mathcal{A})$. Now we can construct $\mathcal{A}^{prop}$. Again from [GKM06, Prop.5.14] we obtain a $\Gamma(\mathcal{A})$-extended CFM $\mathcal{A}^B$ such that $\mathcal{L}_{ext}(\mathcal{A}^B)$ is the set of all $\exists$-B-bounded MSCs over $\Gamma(\mathcal{A})$.

Now we look at the automaton $\mathcal{A}^{prop} \cap \mathcal{A}^B$ which recognizes the language $\mathcal{L}_{ext}(\mathcal{A}^{prop}) \cap \mathcal{L}_{ext}(\mathcal{A}^B)$ and is effectively constructible. The language accepted by this automaton is $\exists$-B-bounded and so we can apply Theorem 17 to get an MSO($\Gamma(\mathcal{A})$) formula such that $\mathcal{L}_{ext}(\mathcal{A}^{prop} \cap \mathcal{A}^B) = \mathcal{L}_{ext}(\varphi)$. Again we can say $\Gamma(\varphi) = \Gamma(\mathcal{A})$ because the intervals occurring in φ come from $\mathcal{A}$ (and for any missing intervals $I \in prop(voc_\alpha(\mathcal{A}))$ we can add them by looking at $\varphi \vee (\alpha(x) \in I \wedge \neg(\alpha(x) \in I))$. This does not change the lan-

guage accepted). Finally using the MSO-TMSO translation we get a TMSO formula φ^t.

Then we have the following sequence of deductions,
$T \in \mathcal{L}_{time}(\mathcal{A}) \implies M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A}^{prop})$ by Lemma 15. By Corollary 10 of Lemma 9, T is $\exists^u$-B-bounded implies that $M^T_{\Gamma(\mathcal{A})}$ is $\exists$-B-bounded. Therefore $M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A}^B)$. Hence $M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A}^{prop} \cap \mathcal{A}^B)$.
But from above we have $\mathcal{L}_{ext}(\mathcal{A}^{prop} \cap \mathcal{A}^B) = \mathcal{L}_{ext}(\varphi)$ and again $M^T_{\Gamma(\varphi)} = M^T_{\Gamma(\mathcal{A})}$. So $M^T_{\Gamma(\varphi)} \in \mathcal{L}_{ext}(\varphi)$, i.e, $M^T_{\Gamma(\varphi)} \models \varphi$. By Lemma 12, this implies $T \models \varphi^t$. Thus we have $\mathcal{L}_{time}(\mathcal{A}) \subseteq \mathcal{L}_{time}(\varphi^t)$.

In the other direction if $T \in \mathcal{L}_{time}(\varphi^t)$, we have by Lemma 12, $M^T_{\Gamma(\varphi)} \in \mathcal{L}_{ext}(\varphi)$. Then again we know that $\mathcal{L}_{ext}(\varphi) = \mathcal{L}_{ext}(\mathcal{A}^{prop} \cap \mathcal{A}^B)$ and that $M^T_{\Gamma(\varphi)} = M^T_{\Gamma(\mathcal{A})}$, so we get $M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A}^{prop} \cap \mathcal{A}^B)$ and in particular $M^T_{\Gamma(\mathcal{A})} \in \mathcal{L}_{ext}(\mathcal{A}^{prop})$. But now we apply Lemma 15 again to obtain $T \in \mathcal{L}_{time}(\mathcal{A})$. So we have shown that $\mathcal{L}_{time}(\varphi^t) \subseteq \mathcal{L}_{time}(\mathcal{A})$.

In conclusion from the given EC-CFM we have constructed a TMSO formula such that the set of T-MSCs accepted accepted is the same. □

8 Message sequence charts with Timing Constraints

In this section we introduce a different approach for timing. Instead of associating time labels to events we will attach them to the message flow. Further instead of a single real number representing the absolute time of occurrence we associate an interval to represent the timing constraint. As an example consider Figure 7 below. The label (0,2] on message from u_1 to v_1 specifies the lower bounds and upper bounds on the delay of message delivery. The label [1,4] from u_1 to u_3 represents the bounds on the delay between the occurrence of event u_1 and u_3 and so on.

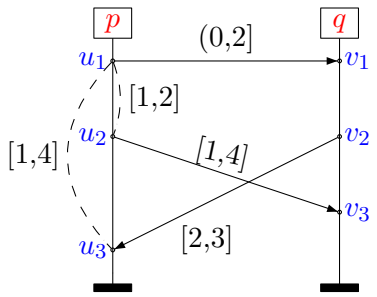

Figure 7: A message sequence chart with timing contraints

Such an extension is better than time-stamped MSCs for two reasons.

i) it preserves the partial order ii) better for specification purposes.

Now let us formally define such an MSC with timing constraints.

Definition 19 (TC-MSC) *Let Δ be a fixed timing alphabet and $\mathcal{I}$ be the set of all intervals over $\mathbb{R}^{\geq 0}$ as usual. Recall that $\prec_\alpha = \{(e, e') \in \prec \mid$ either $e' = \alpha^M(e)$ or $e = \alpha^M(e')\}$. Then, an* MSC with timing constraints *over (Ag, Δ) is a tuple $(E, \preceq, \lambda, \tau)$, where $(E, \preceq, \lambda)$ is an MSC over Ag and $\tau : \bigcup_{\alpha \in \Delta} \prec_\alpha \to \mathcal{I}$ is partial function. The set of all TC-MSCs over (Ag, Δ) is denoted $\mathbb{TCMSC}$.*

This approach is similar to and a generalization of the one adopted by Alur et al [AHP96]. Thus we can use their MSC analysis tool to check consistency of these timing constraints in an MSC. This basic ideas behind this will be sketched in Section 9.

Here we require that any pair of events has at-most one timing constraint and for this reason we use union and not disjoint union in the definition of τ. But we use the timing alphabet since we would like the timing constraints to be user-definable. This results in a generalization of the implementation model as in the approach of [D'S03] where there are only a specific set of clocks.

8.1 Semantics of TC-MSCs in terms of Timed words and T-MSCs.

TC-MSCs can be seen as an abstraction of T-MSCs and timed words. Thus a TC-MSC will correspond to a set of T-MSCs or timed words. Also, given a T-MSC or a timed word we can identify the exact TC-MSC that it corresponds to. We formalize these observations as follows.
Let $M = (E, \preceq, \lambda, \tau)$ be a TC-MSC over (Ag, Δ). If $E = \{e_1, ...e_n\}$, then a timed word $\sigma = (\lambda(e_1), t_1)$ ... $(\lambda(e_n), t_n)$ over *Act* is said to be a *realization* of TC-MSC M if for all $\alpha \in \Delta$ and all pairs $(e_i, e_j) \in \prec_\alpha$, $t_j - t_i \in \tau(e_i, e_j)$ and $e_i \preceq e_j$ implies $i \leq j$. The set of all realizations of TC-MSC is thus a timed language of all such timed words.

A T-MSC $T = (E, \preceq, \lambda, t)$ over Ag is said to be a *realization* of M if $t : E \to \mathbb{R}^{\geq 0}$ is such that for all $\alpha \in \Delta$, and any $(e_1, e_2) \in \prec_\alpha$, $|t(e_2) - t(e_1)| \in \tau(e_1, e_2)$. The set of all T-MSCs that are realizations of a given TC-MSC M is denoted $\mathcal{L}_{time}(M)$.

But given an T-MSC, we have also its semantics in terms of timed words. Then with the above definitions the following proposition is straightforward,

Proposition 20 *the set of timed words that are realizations of a TC-MSC is equal to the set of timed words that are the realizations of all the T-MSCs which are realizations of the given TC-MSC.*

8.2 Semantics of EC-CFMs in terms of TC-MSCs.

For the definition of semantics of an EC-CFM, we can also use TC-MSCs instead of T-MSCs to represent successful runs. This is very similar to the semantics in terms of T-MSCs but has the advantage of being one step more abstract.
Given an EC-CFM $\mathcal{A} = (C, (\mathcal{A}_p)_{p\in Ag}, F)$ and TC-MSC $M = (E, \preceq, \lambda, \tau)$, let $\rho : E \to \bigcup_{p\in Ag} Q_p$ be a map labeling each event of the TC-MSC with a state of a constituent finite state machine of the EC-CFM. We define ρ^- as before. Then we say that ρ is a *run* of $\mathcal{A}$ on TC-MSC M if for $e \in S$ and $e' \in R$ with $\lambda(e) = p!q$ and $Msg(e) = e'$, there are $g, g' : \Delta \to \mathcal{I}$ and a control message $c \in C$ s.t

$$\rho^-(e) \xrightarrow{p!q,g,c}_{Ag(e)} \rho(e) \quad \text{and} \quad \rho^-(e') \xrightarrow{q?p,g',c}_{Ag(e')} \rho(e')$$

with $\tau(e, \alpha^M(e)) \subseteq g(\alpha)$ or $\tau(\alpha^M(e), e) \subseteq g(\alpha)$, for all $\alpha \in \Delta$ s.t e is in the domain of α^M (note that only one of the two above cases is possible), and similarly for e', $\tau(e', \alpha^M(e')) \subseteq g'(\alpha)$ or $\tau(\alpha^M(e'), e') \subseteq g'(\alpha)$, for all $\alpha \in \Delta$ s.t e' is in the domain of α^M. The definition of successful runs and acceptance criterion are as before and we denote by $\mathcal{L}_{TC}(\mathcal{A})$, the set of TC-MSCs that are accepted by $\mathcal{A}$.

8.3 Semantics of TMSOs in terms of TC-MSCs.

Again as in the case of EC-CFMs we can interpret TMSOs over TC-MSCs instead of T-MSCs. The definition of the semantics of a TMSO formula φ over a TC-MSC $M = (E, \preceq, \lambda, \tau)$ is exactly the same as of φ over a T-MSC $T = (E, \preceq, \lambda, t)$ except that in the satisfaction relation we define the timing predicate as follows, $M, \mathbb{I} \models \alpha(x) \in I$ if $\mathbb{I}(x)$ is in the domain of α^M and $\tau(\mathbb{I}(x), \alpha^M(\mathbb{I}(x))) \subseteq I$ or $\tau(\alpha^M(\mathbb{I}(x)), \mathbb{I}(x)) \subseteq I$ or else $\mathbb{I}(x)$ is not in the domain of α^M and $I = [\bot, \bot]$. Again for sentences φ, we have $\mathcal{L}_{TC}(\varphi)$ is the set of TC-MSCs that satisfy φ.

8.4 The relationship between TC-MSCs and T-MSCs

We will devote the rest of this section to further investigation of the relationship between TC-MSCs and T-MSCs in the context of satisfiability of logical formulas. TC-MSCs are ideal for specification purposes and T-MSCs are actual realizations which are useful for formal theorem proving (eg.,Theorem 18), so it is interesting to look into the relationship between them and see if we can have a one-to-one correspondence between them.

In fact, one direction is easy to see, i.e, if we have a TC-MSC satisfying a TMSO then it is clear that all its timed realizations also satisfy the formula.

Proposition 21 *Given TC-MSC M, TMSO φ, $M \models \varphi$ implies $T \models \varphi$ for all $T \in \mathbb{TMSC}$ such that T is a realization of M.*

Proof. Let $M = (E, \preceq, \lambda, \tau)$, $T = (E, \preceq, \lambda, t)$ be a realization of M, i.e, for all $e_1, e_2 \in E$ such that $(e_1, e_2) \in \prec_\alpha$ for some $\alpha \in \Delta$, $t(e_1) - t(e_2) \in \tau(e_1, e_2)$. The proof is of course by structural induction on φ over an interpretation $\mathbb{I}$. The only interesting case is that of the timing predicate, for all others the definitions of satisfaction relation coincide.

So $M, \mathbb{I} \models \alpha(x) \in I$ implies either $\mathbb{I}(x)$ is in domain of α^M or it is not in the domain. In the former case we have $\tau(\mathbb{I}(x), \alpha^M(\mathcal{I}(x)) \subseteq I$ (wlog., we omit the symmetric case of $\tau(\alpha^M(\mathcal{I}(x), \mathbb{I}(x)) \subseteq I$). This in turn means that $\mathbb{I}(x)$ is in the domain of α^T ($\alpha^M = \alpha^T$ since they share the same underlying MSC). But from the fact that T is realization of M we get that $|t(\mathbb{I}(x)) - t(\alpha^T(\mathbb{I}(x)))| \in \tau(\mathbb{I}(x), \alpha^T(\mathbb{I}(x))) \subseteq I$. In the latter case if $\mathbb{I}(x)$ is not in the domain of α^M, this implies that $\mathbb{I}(x)$ is not in the domain of α^T and that $I = [\bot, \bot]$. Therefore we conclude that if $\mathbb{I}(x)$ is in the domain of α^T $|t(\mathbb{I}(x)) - t(\alpha^T(\mathbb{I}(x)))| \in I$ and if it is not in the domain we have $I = [\bot, \bot]$. But this is the definition of $T, \mathbb{I} \models \alpha(x) \in I$. Thus we are done. And by structural induction we have proved the proposition. □

But the other direction is not so simple. In particular even if every T-MSC realization of a TC-MSC satisfies a formula the TC-MSC itself may not satisfy it.

Example 22 *Let Ag, Δ be fixed such that the message clock $Msg \in \Delta$ as in Example 1. Then suppose we are given the TMSO formula,*

$$\varphi = \exists x(Msg(x) \in [2, 3]) \vee (Msg(x) \in [3, 5])$$

then the TC-MSC M of Figure 8 does not satisfy it though all its timed realizations do satisfy it. This can be seen since event e of the TC-MSC has to have the exact same interval as in the formula, whereas for any T-MSC $T \in \mathcal{L}_{time}(M)$, we just need the difference between time stamp of e and e' to be in any interval specified by the formula.

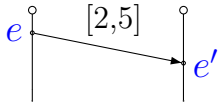

Figure 8: TC-MSC M

In fact we can note further that two formulas which have the same set of timed realizations (and thus maybe thought of intuitively as equivalent) may have different sets of TC-MSCs. Just consider φ as above and

$$\varphi' = \exists x(Msg(x) \in [2, 5])$$

Then, $\mathcal{L}_{time}(\varphi) = \mathcal{L}_{time}(\varphi')$ as a set of T-MSCs, but $M \in \mathcal{L}_{TC}(\varphi')$ and as seen above $M \notin \mathcal{L}_{TC}(\varphi)$.

As these above examples show TC-MSCs are not robust enough to capture realizations of formulae. However we can obtain the following "weaker" correspondence in the reverse direction,

Proposition 23 *Given TC-MSC M and TMSO φ, if for all $T \in \mathcal{L}_{time}(M)$, $T \models \varphi$ then we can compute a nonempty finite set of TC-MSCs $\mathcal{L}$ such that*

1. *$T \in \mathcal{L}_{time}(M')$ for some $M' \in \mathcal{L} \implies T \in \mathcal{L}_{time}(M)$.*
2. *$T \in \mathcal{L}_{time}(M) \implies T \in \mathcal{L}_{time}(M')$ for exactly one M' such that $M' \in \mathcal{L}$.*
3. *for all $M' \in \mathcal{L}$, $M' \models \varphi$*

Proof. Given TC-MSC $M = (E, \preceq, \lambda, \tau)$ and TMSO φ. Recall that $voc_\alpha(\varphi)$ is the set of intervals $I \in \mathcal{I}$ for which there exists a sub-formula of φ of the form $\alpha(x) \in I$. Let $voc_\alpha(M)$ be the set of intervals of α mentioned in the TC-MSC, i.e, $\{\tau(e, e') \mid (e, e') \in \prec_\alpha \text{ for some } e, e' \in E\}$. Note that both $voc_\alpha(\varphi)$ and $voc_\alpha(M)$ are finite for all $\alpha \in \Delta$. Now, for each $\alpha \in \Delta$, let $\mathcal{I}_\alpha = prop(voc_\alpha(\varphi) \cup voc_\alpha(M))$ be the canonical proper interval set which refines it, as defined in Section 4.5.

Now we define our set of TC-MSCs $\mathcal{L}$ as follows: $\mathcal{L} = \{M' \in \mathbb{TCMSC} \mid M' = (E, \preceq, \lambda, \tau')$ such that for all $\alpha \in \Delta$ and all pairs of events $e_1, e_2 \in E$ with $(e_1, e_2) \in \prec_\alpha$, we have $\tau'(e_1, e_2) \in \mathcal{I}_\alpha$ and $\tau'(e_1, e_2) \subseteq \tau(e_1, e_2)\}$. Now with this definition of $\mathcal{L}$ we show that properties 1, 2 and 3 hold. For any TC-MSC $M' = (E, \preceq, \lambda, \tau') \in \mathcal{L}$, let $T = (E, \preceq, \lambda, t)$ denote an arbitrary element of $\mathcal{L}_{time}(M')$.

1. For any $M' \in \mathcal{L}$, $T \in \mathcal{L}_{time}(M')$ implies that for all α, $(e_1, e_2) \in \prec_\alpha$ $t(e_2) - t(e_1) \in \tau'(e_1, e_2)$. But from the definition of $\mathcal{L}$ we have that for all α, $(e_1, e_2) \in \prec_\alpha$, $\tau'(e_1, e_2) \subseteq \tau(e_1, e_2)$. This implies that for all $\alpha, (e_1, e_2) \in \prec_\alpha$, $t(e_2) - t(e_1) \in \tau(e_1, e_2) \implies T \in \mathcal{L}_{time}(M)$.

2. $T \in \mathcal{L}_{time}(M) \implies$ for all $\alpha, (e_1, e_2) \in \prec_\alpha, t(e_2) - t(e_1) \in \tau(e_1, e_2)$. But for a fixed α and pair of events $(e_1, e_2) \in \prec_\alpha$, by properness of interval set $\mathcal{I}_\alpha$, $t(e_2) - t(e_1) \in I$ for exactly one $I \in \mathcal{I}_\alpha$. Define $\tau'(e_1, e_2) = I$. We do this for all $\alpha \in \Delta$ and all pairs of events $(e_1, e_2) \in \prec_\alpha$. This defines M' completely. And now it is easy to see that M' thus defined belongs to $\mathcal{L}$ and also that $T \in \mathcal{L}_{time}(M')$. Further M' is unique because if this is not the case, we will have that for some $\alpha \in \Delta$ and some pair of events $(e_1, e_2) \in \mathcal{I}_\alpha$, $t(e_2) - t(e_1) \in I$ and $t(e_2) - t(e_1) \in I'$ with $I, I' \in \mathcal{I}_\alpha$ which is a contradiction of properness of $\mathcal{I}_\alpha$.

3. Here we want to show $M' \models \varphi$ for all $M' \in \mathcal{L}$. We will show this by proving a slightly more general statement. In particular we prove that:

Fix an $M' \in \mathcal{L}$. Then for all $T \in \mathcal{L}_{time}(M')$, interpretations $\mathbb{I}$ and TMSO formulae φ we have, If $T, \mathbb{I} \models \varphi$, then $M', \mathbb{I} \models \varphi$.
The proof is by structural induction on φ. Note that for M, M' and any T the interpretation assigns the same values to variables and that their underlying triple $(E, \preceq, \lambda)$ is the same. Now, let $\varphi = P_a(x)$. Then $T, \mathbb{I} \models P_a(x)$ implies that $\lambda(\mathbb{I}(x)) = a$. But since λ is same for T and M', $M', \mathbb{I} \models P_a(x)$ Similarly the cases $\varphi = x \in X$ and $x \leq y$ are trivial from definitions.

The interesting case is $\varphi = \alpha(x) \in I$. Let $T \in \mathcal{L}_{time}(M')$ be such that $T, \mathbb{I} \models \alpha(x) \in I$. Suppose $\mathbb{I}(x)$ is not in the domain of α^T, then it is not in the domain of $\alpha^{M'}$ either and $I = [\perp, \perp]$, therefore $M', \mathbb{I} \models \varphi$. If not, we have $\mathbb{I}(x)$ is in domain of α^T and so $\mathbb{I}(x)$ is in domain of $\alpha^{M'}$. Now, $M' \in \mathcal{L}$ implies that either $\tau'(\mathbb{I}(x), \alpha^{M'}(\mathbb{I}(x))) \in \mathcal{I}_\alpha$ or $\tau'(\alpha^{M'}(\mathbb{I}(x)), \mathbb{I}(x)) \in \mathcal{I}_\alpha$. Denote this interval by $\widehat{I}$ so that $\widehat{I} \in \mathcal{I}_\alpha$. But $I \in voc_\alpha(\varphi)$ and $\mathcal{I}_\alpha$ is a proper interval set refining $I \in voc_\alpha(\varphi)$ (in fact it refines a bigger set). This means that either $\widehat{I} \subseteq I$ or $\widehat{I} \cap I = \emptyset$. But $T, \mathbb{I} \models \alpha(x) \in I$ implies $|t(\mathbb{I}(x)) - t(\alpha^T(\mathbb{I}(x)))| \in I$ and $T \in \mathcal{L}_{time}(M')$ implies $|t(\mathbb{I}(x) - t(\alpha^{M'}(\mathbb{I}(x)))| \in \widehat{I}$. Thus they have a common point and so the intersection is non empty. This implies $\widehat{I} \subseteq I$. Which implies that $M', \mathbb{I} \models \varphi$. Thus we are done with this case too.

The remaining cases are straightforward structural induction deductions.

□

We can also observe from the above that the formula φ cannot "distinguish" between realizations of one particular such M'. In other words, for all $T_1, T_2 \in \mathcal{L}_{time}(M')$, $T_1 \models \varphi \iff T_2 \models \varphi$. Thus these TC-MSCs are in some sense good representatives of the timed realizations with respect to a Γ.

9 Some elementary implementable architectures

In this section we have a brief glance at some satisfiability and model checking issues for MSCs with timing. Since the message sequence charts are used to early protocol design, these problems are of vital interest as they allow detection of possible design failures at this stage. Such an early specification of a protocol can have several deficiencies either with respect to partial order of events (e.g, race conditions) or violation of user-defined properties specified in logics with or without a timed view. But in general these problems are either undecidable or of very high complexity even for the untimed version.

The realizability of a single MSC in an actual system execution corresponds most commonly to the absence of certain "race" conditions. This means that two events occur in one (visual) order in the MSC but can be shown to occur in the opposite order during the execution. In the paper of [AHP96] we have an algorithm for analysing MSCs for such conditions which works by computing the transitive closure. This approach would also extend to MSCs with time stamps. For the case of MSCs with timing constraints we additionally need to check that these constraints are not invalidated. This result is also given in the above mentioned paper. It works by constructing an integer weighted graph corresponding to the TC-MSC and reducing the problem of realizability to a problem of searching for negative cost cycles in the graph.

Now we move to investigating the satisfiability of timed formulas over timed MSCs.

First we observe that the timing alphabet plays a vital role in determining this nature of implementability of corresponding models. This is because the way we define the timing alphabet and the maps within determines the power of the clocks and also their expressiveness in the logic. In addition we also have the choice of existential or universal boundedness. Of course all this works after fixing a finite set of processes. We recall that the timing alphabet Δ is defined as a finite alphabet of symbols each of which is interpreted as a partial function $\alpha : E \to E$ for each MSC.

The problem of satisfiability of a TMSO in its full generality is undecidable:
Problem 1: Let Ag and Δ be any fixed set of processes and a timing alphabet,

- Given : $\varphi \in \mathrm{TMSO}(Act)$
- Question: Is there $T \in \mathbb{TMSC}$ such that $T \models \varphi$?

Theorem 24 *Problem 1 is undecidable.*

Proof. Follows from the untimed case, since by [GKM06], the problem is undecidable even in that case. □

It is clear that many questions here remain unresolved and we mention a few of them in the next section.

10 Conclusion and Future work

We have studied various approaches to introduce timing for formal specification using MSCs. We have seen that time-stamped or timed MSCs are a useful formalism for describing timing in communicating systems. We

showed here the equivalence of automata and logic models for the specification of existentially bounded sets of timed MSCs in Theorem 18. For this we introduced EC-CFMs as a timed distributed automaton model with clocks to express their timing constraints. For the logic we used TMSO having interval predicates. The key idea of the proof was to "untime" the structures by extending the alphabet with intervals (as done by D'Souza [D'S03])to get an untimed structure. We do this for all three relevant structures, i.e, Timed MSCs, TMSO formulae and EC-CFMs and show that this preserves the semantics in the sense of Lemmas 11,12 and 15. Thus we are now in the untimed setting of [GKM06] except that we have extended alphabets. But this does not affect their result and we are able to use this as stated in Theorem 17 to conclude the constructive equivalence between automata and logic over $\exists^u$-bounded timed MSCs.

Further we studied another formalism for timing in MSCs, TC-MSCs, which are more pliable to specification and more natural for practical reasons. We compared this approach to the previous one to show both the similarity and the differences. In particular we see why this is not as ideal for describing realizations and for the equivalence as timed MSCs. Nevertheless, we give a partial correspondence between these formalisms. Finally we had a preliminary glance at some satisfiability and implementability issues. We have seen how the problem in its general version remains undecidable.

The major *open questions* concern the implementability issues for both single and sets of timed MSCs.
Currently we are in the process of proving the existence of a decidable model which is highly restricted in allowable timing conditions in the setting of universally bounded T-MSCs. We also believe that this would generalize to the existentially bounded case but this is yet to be shown.
Further we are left with many questions on less restrictive and more interesting models of timing and the implementability issues concerning them. It would also be interesting to look at other timing alphabets which could show clearly (or give evidence to the absence of) the difference between timed and untimed models of communicating systems.
In our approach we have used automata models with clocks since we had a good logical characterizations for this, but one could also generalize this approach for other classes of timed automata which are both decidable and which have a decent logical characterization.
Also we have the question of practically implementable restrictions of EC-CFMs would. In general in an EC-CFM even with restricted models of timing the number of clocks could be infinite if the channels are unbounded. It would be nice to look at the restriction to finitely many clocks and try for an equivalent logical formalism.

References

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[AFH99] Rajeev Alur, Limor Fix, and Thomas A. Henzinger. Event-clock automata: A determinizable class of timed automata. *Theoretical Computer Science*, 211(1-2):253–273, 1999.

[AHP96] Rajeev Alur, Gerard J. Holzmann, and Doron Peled. An analyser for message sequence charts. In *Proceedings of the 2nd International Workshop on Tools and Algorithms for Construction and Analysis of Systems (TACAS 1996), Passau, Germany*, pages 35–48, 1996.

[BAL97] Hanêne Ben-Abdallah and Stefan Leue. Timing constraints in message sequence chart specifications. In *FORTE*, pages 91–106, 1997.

[Büc60] J. Büchi. Weak second order logic and finite automata. *Z. Math. Logik, Grundlag. Math.*, 5:66–62, 1960.

[BZ83] D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2), 1983.

[DR95] V. Diekert and G. Rozenberg, editors. *The Book of Traces*. World Scientific, Singapore, 1995.

[D'S03] Deepak D'Souza. A logical characterisation of event clock automata. *International Journal of Foundations of Computer Science*, 14(4):625–640, 2003.

[Elg61] C. C. Elgot. Decision problems of finite automata design and related arithmetics. *Trans. Amer. Math. Soc.*, 98:21–52, 1961.

[GKM06] B. Genest, D. Kuske, and A. Muscholl. A Kleene theorem and model checking algorithms for existentially bounded communicating automata. *Information and Computation*, 204(6):920–956, 2006.

[HMK+05] J. G. Henriksen, M. Mukund, K. Narayan Kumar, M. Sohoni, and P. S. Thiagarajan. A theory of regular MSC languages. *Information and Computation*, 202(1):1–38, 2005.

[ITU98] ITU-TS Recommendation Z.120anb: Formal Semantics of Message Sequence Charts, 1998.

[ITU99] ITU-TS Recommendation Z.120: Message Sequence Chart 1999 (MSC99), 1999.