Revisiting timed logics with automata modalities

HO, HM <http://orcid.org/0000-0003-0387-4857>

Revisiting Timed Logics with Automata Modalities

Hsi-Ming Ho
University of Cambridge
United Kingdom
hsi-ming.ho@cl.cam.ac.uk

ABSTRACT

It is well known that (timed) ω-regular properties such as 'p holds at every even position' and 'p occurs at least three times within the next 10 time units' cannot be expressed in Metric Interval Temporal Logic (MITL) and Event Clock Logic (ECL). A standard remedy to this deficiency is to extend these with modalities defined in terms of automata. In this paper, we show that the logics $\mathsf{EMITL}_{0,\infty}$ (adding *non-deterministic finite automata* modalities into the fragment of MITL with only lower- and upper-bound constraints) and EECL (adding automata modalities into ECL) are already as expressive as EMITL (full MITL with automata modalities). In particular, the satisfiability and model-checking problems for $\mathsf{EMITL}_{0,\infty}$ and EECL are PSPACE-complete, whereas the same problems for EMITL are EXPSPACE-complete. We also provide a simple translation from $\mathsf{EMITL}_{0,\infty}$ to *diagonal-free* timed automata, which enables practical satisfiability and model checking based on off-the-shelf tools.

CCS CONCEPTS

•Theory of computation → Timed and hybrid models; Logic and verification; Verification by model checking;

KEYWORDS

metric interval temporal logic, timed automata, model checking

1 INTRODUCTION

Timed logics. In the context of real-time systems verification, it is natural and desirable to add *timing constraints* to *Linear Temporal Logic* (LTL) [42] to enable reasoning about timing behaviours of such systems. For instance, one may write $\varphi_1 \,\mathbf{U}_I\, \varphi_2$ to assert that φ_1 holds until a 'witness' point where φ_2 holds, and the time difference between now and that point lies within the *constraining interval* I. The resulting logic, *Metric Temporal Logic* (MTL) [32], can be seen as a fragment of *Monadic First-Order Logic of Order and Metric* (FO[<, +1]) [4], the timed counterpart of the classical *Monadic First-Order Logic of Order* (FO[<]). There are, nonetheless, some loose ends in this analogy. For instance, while LTL is as expressive as FO[<] [22, 29], it is noted early on that certain 'non-local' timing properties in FO[<, +1], albeit being very simple, cannot be expressed in timed temporal logics like MTL [5]. As a concrete example, the property 'every p-event is followed by a q-event and, later, an r-event within the next 10 time units', written as the FO[<, +1] formula

$$\forall x \left(p(x) \Rightarrow \exists y \left(q(y) \wedge \exists z \left(r(z) \wedge x \leq y \leq z \leq x + 10\right)\right)\right) \qquad (1)$$

is not expressible in MTL—indeed, no 'finitary' extension of MTL can be *expressively complete* for FO[<, +1] [28].[1] A more serious practical concern is that the satisfiability problem for MTL is undecidable [4, 40]. For this reason, research efforts have been focused on fragments of MTL with decidable satisfiability, most notably *Metric Interval Temporal Logic* (MITL), the fragment of MTL in which 'punctual' constraining intervals are not allowed [3]. In particular, MITL formulae can be effectively translated into *timed automata* (TAs) [2], giving practical EXPSPACE decision procedures for its *satisfiability* and *model-checking* problems [16–18].

Automata modalities. It is well known that properties that are necessarily *second order* (e.g., 'p holds at all even positions') cannot be expressed in LTL or MITL. Fortunately, it is possible to add *automata modalities* into LTL at no additional computational cost [46, 50]. In timed settings, the logic obtained from MITL by adding *time-constrained* automata modalities defined by nondeterministic finite automata (NFAs) is called *Extended Metric Interval Temporal Logic* (EMITL) [48]. From a theoretical point of view, EMITL is a *fully decidable* formalism (i.e. constructively closed under all Boolean operations and with decidable satisfiability [26]) whose class of timed languages strictly contains that of MITL and Büchi automata.[2] In practice, it can be argued that automata modalities are natural, easy-to-use extensions of the usual MITL modalities. They also allow properties like (1), which often emerge in application domains like healthcare and automotive engineering, to be written as specifications.

Example 1.1 ([1]). Discrimination algorithms are implemented in implantable cardioverter defibrillators (ICDs) to detect potentially dangerous heartbeat patterns. As a simple example, one may want to check whether *the number of heartbeats in one minute is between* 120 *and* 150. This can be expressed as the CTMITL [33] formula $\mathbf{C}^{\geq 120}_{[0,59]}\, p \wedge \mathbf{C}^{\leq 150}_{[0,59]}\, p$ where p denotes a peak in the cardiac signal. The *counting modalities* $\mathbf{C}^{\sim k}_I$ (where $0 \in I$, which is the case here), as well as (1), be expressed straightforwardly in terms of automata.

Example 1.2 (adapted from [25]). In autonomous driving, one may want to specify that *a car overtaking another from the left must be done in* 10 *seconds*. Suppose the lane on the left is empty and the events are sampled sufficiently frequently (say 5ms), this can be expressed as the EMITL formula $\mathcal{A}_{[0,10]}(\texttt{TTC > 4}, \dots)$ (see Fig. 1 and Fig. 2) where `TTC` is the time to collision, `dist` is the longitudinal distance between the two vehicles, and `to_left`, `to_right` are the actions for merging to the left/right lane—these are taken immediately after `TTC <= 4` and `dist >= 5`, respectively.

Compared with LTL and MITL, however, translating EMITL into TAs is considerably more challenging. The original translation by Wilke [48] is non-elementary and thus not suitable for practical

[1] (1) can, however, be expressed in MTL if the *continuous* semantics of the logic is adopted or past modalities are allowed; see [14] for details.

[2] A very recent paper of Krishna, Madnani, and Pandya [35] showed that this class admits some alternative characterisations (namely, a syntactic fragment of OCATAs and a timed monadic second-order logic).

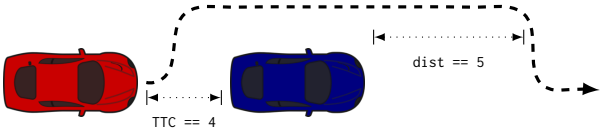

Fig. 1: The red car overtakes the blue car from the left.

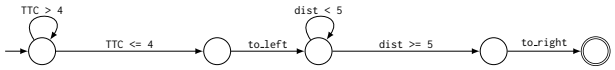

Fig. 2: $\mathcal{A}$ in Example 1.2.

purposes. Krishna, Madnani, and Pandya [34] showed that any EMITL formula can be encoded into an MITL formula of doubly exponential size (which can then be translated into a TA), but this does not match the EXPSPACE lower bound inherited from MITL. More recently, Ferrère [21] proposed an asymptotically optimal construction from MIDL (*Metric Interval Dynamic Logic*, which is strictly more expressive and subsumes EMITL) formulae to TAs, but it is very complicated and relies heavily on the use of *diagonal constraints* (i.e. comparison between clocks) which are, in general, not preferred in practice [12, 15, 23] and not well-supported by existing model checkers.[3]

Contributions. We consider a simple fragment of EMITL, which we call $\mathsf{EMITL}_{0,\infty}$, obtained by allowing only lower- and upper-bound constraining intervals (e.g., $[0, a)$ and (b, ∞)) and EECL [43] (adding automata modalities to *Event Clock Logic* ECL). The satisfiability and model-checking problems for $\mathsf{EMITL}_{0,\infty}$ and EECL are much cheaper than that of EMITL (PSPACE-complete vs EXPSPACE-complete). Moreover, we show that they are already as expressive as full EMITL—this is in sharp contrast with the situation for 'vanilla' $\mathsf{MITL}_{0,\infty}$/ECL and MITL, where the latter is strictly more expressive when interpreted over timed words [26, 43]—making them *expressive* yet *tractable* real-time specification formalisms. We then show that $\mathsf{EMITL}_{0,\infty}$ admits a much simpler translation into TAs. Specifically, by effectively decoupling the timing and operational aspects of automata modalities, overlapping obligations imposed by a single automaton subformula can be handled in a purely fifo manner with a set of *sub-components* (each of which is a simple one-clock TA with a polynomial-sized symbolic representation), avoiding the use of diagonal constraints altogether.[4] This makes our construction better suited to be implemented to work with existing highly efficient algorithmic back ends (e.g., Uppaal [11] and LTSmin [30]).

Related work. The idea of extending LTL to capture the full class of ω-regular languages dates back to the seminal works of Clarke, Sistla, Vardi, and Wolper [45, 46, 49, 50] in the early 1980s. In particular, it is shown that LTL with NFA modalities—which essentially underlies various industrial specification languages like ForSpec [6] and PSL [20]—are expressively equivalent to Büchi automata, yet the model-checking and satisfiability problems remain PSPACE-complete, same as LTL.[5] Our approach generalises the construction in [50] in the case of finite acceptance.

Henzinger, Raskin, and Schobbens [26, 43] proved a number of analogous results in timed settings; in particular, they showed that in the *continuous* semantics (i.e. over finitely variable *signals*), (i) $\mathsf{MITL}_{0,\infty}$ and ECL are as expressive as MITL, and (ii) the fragment of EMITL with *unconstrained* automata modalities is as expressive as *recursive event-clock automata*, and the verification problems for this fragment can be solved in EXPSPACE. Our results can be seen as counterparts in the *pointwise* semantics (i.e. over *timed words*).

Besides satisfiability and model checking, extending timed logics with automata or regular expressions is also a topic of great interest in *runtime verification*. Basin, Krstić, and Traytel [10] showed that MTL with time-constrained regular-expression modalities admits an efficient runtime monitoring procedure in a pointwise, integer-time setting. A very recent work of Ničković, Lebeltel, Maler, Ferrère, and Ulus [39] considered a similar extension of MITL with *timed regular expressions* (TRE) [7, 8] in the context of monitoring and analysis of Boolean and real-valued signals.

2 TIMED LOGICS AND AUTOMATA

Timed languages. A *timed word* over a finite alphabet Σ is an infinite sequence of *events* $(\sigma_i, \tau_i)_{i\geq 1}$ over $\Sigma \times \mathbb{R}_{\geq 0}$ with $(\tau_i)_{i\geq 1}$ a non-decreasing sequence of non-negative real numbers such that for each $r \in \mathbb{R}_{\geq 0}$, there is some $j \geq 1$ with $\tau_j \geq r$ (i.e. we require all timed words to be '*non-Zeno*'). We denote by $T\Sigma^\omega$ the set of all timed words over Σ. A *timed language* is a subset of $T\Sigma^\omega$.

Extended timed logics. A *non-deterministic finite automaton* (NFA) over Σ is a tuple $\mathcal{A} = \langle \Sigma, S, s_0, \Delta, F\rangle$ where S is a finite set of locations, $s_0 \in S$ is the initial location, $\Delta \subseteq S\times\Sigma\times S$ is the transition relation, and F is the set of final locations. We say that $\mathcal{A}$ is *deterministic* (a DFA) iff for each $s \in S$ and $\sigma \in \Sigma$, $|\{(s, \sigma, s') \mid (s, \sigma, s') \in \Delta\}| \leq 1$. A *run* of $\mathcal{A}$ on $\sigma_1 \dots \sigma_n \in \Sigma^+$ (without loss of generality, we only consider runs of automata modalities over *nonempty* finite words in this paper) is a sequence of locations $s_0 s_1 \dots s_n$ where there is a transition $(s_i, \sigma_{i+1}, s_{i+1}) \in \Delta$ for each i, $0 \leq i < n$. A run of $\mathcal{A}$ is *accepting* iff it ends in a final location. A finite word is *accepted* by $\mathcal{A}$ iff $\mathcal{A}$ has an accepting run on it. We denote by $[\![\mathcal{A}]\!]$ the set of finite words accepted by $\mathcal{A}$.

Extended Metric Interval Temporal Logic (EMITL) formulae over a finite set of atomic propositions AP are generated by

$$\varphi := \top \mid p \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \mathcal{A}_I(\varphi_1, \dots, \varphi_n)$$

where $p \in \mathsf{AP}$, $\mathcal{A}$ is an NFA over the n-ary alphabet $\{1, \dots, n\}$, and $I \subseteq \mathbb{R}_{\geq 0}$ is a non-singular interval with endpoints in $\mathbb{N}_{\geq 0} \cup \{\infty\}$.[6] As usual, we omit the subscript I when $I = [0, \infty)$ and write pseudo-arithmetic expressions for lower or upper bounds, e.g., '< 3' for $[0, 3)$. We also omit the arguments $\varphi_1, \dots, \varphi_n$ and simply write $\mathcal{A}_I$, if clear from the context. Following [4, 5, 41, 48], we consider the pointwise semantics of EMITL and interpret formulae over

[3] It is possible to obtain a diagonal-free TA from an EMITL formula by first applying the construction in [21] and then removing the diagonal constraints [13]. This, however, is expensive and difficult to implement.

[4] For simplicity we focus on logics with only future modalities, but our results readily carry over to the versions with both future and past modalities, thanks to the compositional nature of our construction (cf. e.g., [31, 38]).

[5] There are other ways to extend LTL to achieve ω-regularity, e.g., adding monadic second-order quantifiers (QPTL [46]) or least/greatest fixpoints (μLTL [9, 47]). These formalisms unfortunately suffer from higher complexity or less readable syntax.

[6] For notational simplicity, we will occasionally use $\varphi_1, \dots, \varphi_n$ directly as transition labels (instead of $1, \dots, n$).

timed words: given an EMITL formula φ over AP, a timed word $\rho = (\sigma_1, \tau_1)(\sigma_2, \tau_2)\dots$ over $\Sigma_{\mathsf{AP}} = 2^{\mathsf{AP}}$ and a *position* $i \geq 1$,

- $(\rho, i) \models \top$;
- $(\rho, i) \models p$ iff $p \in \sigma_i$;
- $(\rho, i) \models \varphi_1 \wedge \varphi_2$ iff $(\rho, i) \models \varphi_1$ and $(\rho, i) \models \varphi_2$;
- $(\rho, i) \models \neg\varphi$ iff $(\rho, i) \not\models \varphi$;
- $(\rho, i) \models \mathcal{A}_I(\varphi_1, \dots, \varphi_n)$ iff there exists $j \geq i$ such that (i) $\tau_j - \tau_i \in I$ and (ii) there is an accepting run of $\mathcal{A}$ on $a_i \dots a_j$ where $a_\ell \in \{1, \dots, n\}$ and $(\rho, \ell) \models \varphi_{a_\ell}$ for each ℓ, $i \leq \ell \leq j$.[7]

The other Boolean operators are defined as usual: $\bot \equiv \neg\top$, $\varphi_1 \vee \varphi_2 \equiv \neg(\neg\varphi_1 \wedge \neg\varphi_2)$, and $\varphi_1 \Rightarrow \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2$. We also define the dual automata modalities $\tilde{\mathcal{A}}_I(\varphi_1, \dots, \varphi_n) \equiv \neg\mathcal{A}_I(\neg\varphi_1, \dots, \neg\varphi_n)$. With the dual automata modalities, we can transform every EMITL formula φ into *negative normal form*, i.e. an EMITL formula using only atomic propositions, their negations, and the operators $\vee$, $\wedge$, $\mathcal{A}_I$, and $\tilde{\mathcal{A}}_I$. It is easy to see that the standard MITL 'until' $\varphi_1 \,\mathbf{U}_I\, \varphi_2$ can be defined in terms of automata modalities. We also use the usual shortcuts like $\mathbf{F}_I\, \varphi \equiv \top \,\mathbf{U}_I\, \varphi$, $\mathbf{G}_I\, \varphi \equiv \neg\, \mathbf{F}_I \neg\varphi$, and $\varphi_1 \,\mathbf{R}_I\, \varphi_2 \equiv \neg\big((\neg\varphi_1) \,\mathbf{U}_I\, (\neg\varphi_2)\big)$. We say that ρ *satisfies* φ (written $\rho \models \varphi$) iff $(\rho, 1) \models \varphi$, and we write $[\![\varphi]\!]$ for the timed language of φ, i.e. the set of all timed words satisfying φ. $\mathsf{EMITL}_{0,\infty}$ is the fragment of EMITL where all constraining intervals I must be lower or upper bounds (e.g., < 3 or ≥ 5). *Extended Event Clock Logic* (EECL) is the fragment of EMITL where $\mathcal{A}_I$ is replaced by a more restricted 'event-clock' counterpart:

- $(\rho, i) \models \overset{\triangleright}{\mathcal{A}}_I\, (\varphi_1, \dots, \varphi_n)$ iff (i) there is a *minimal* position $j \geq i$ such that $\mathcal{A}$ has an accepting run on $a_i \dots a_j$ where $a_\ell \in \{1, \dots, n\}$ and $(\rho, \ell) \models \varphi_{a_\ell}$ for each ℓ, $i \leq \ell \leq j$; and (ii) j satisfies $\tau_j - \tau_i \in I$.

Timed automata. Let X be a finite set of *clocks* ($\mathbb{R}_{\geq 0}$-valued variables). A *valuation* v for X maps each clock $x \in X$ to a value in $\mathbb{R}_{\geq 0}$. We denote by $\mathbf{0}$ the valuation that maps every clock to 0, and we write the valuation simply as a value in $\mathbb{R}_{\geq 0}$ when X is a singleton. The set $\mathcal{G}(X)$ of *clock constraints* g over X is generated by $g := \top \mid g \wedge g \mid x \bowtie c$ where $\bowtie \in \{\leq, <, \geq, >\}$, $x \in X$, and $c \in \mathbb{N}_{\geq 0}$. The satisfaction of a clock constraint g by a valuation v (written $v \models g$) is defined in the usual way, and we write $[\![g]\!]$ for the set of valuations v satisfying g. For $t \in \mathbb{R}_{\geq 0}$, we let $v + t$ be the valuation defined by $(v + t)(x) = v(x) + t$ for all $x \in X$. For $\lambda \subseteq X$, we let $v[\lambda \leftarrow 0]$ be the valuation defined by $(v[\lambda \leftarrow 0])(x) = 0$ if $x \in \lambda$, and $(v[\lambda \leftarrow 0])(x) = v(x)$ otherwise.

A *timed automaton* (TA) over Σ is a tuple $\mathcal{A} = \langle \Sigma, S, s_0, X, \Delta, \mathcal{F} \rangle$ where S is a finite set of locations, $s_0 \in S$ is the initial location, X is a finite set of clocks, $\Delta \subseteq S \times \Sigma \times \mathcal{G}(X) \times 2^X \times S$ is the transition relation, and $\mathcal{F} = \{F_1, \dots, F_n\}$, with $F_i \subseteq S$ for all i, $1 \leq i \leq n$, is the set of sets of final locations.[8] We say that $\mathcal{A}$ is *deterministic* (a DTA) iff for each $s \in S$ and $\sigma \in \Sigma$ and every pair of transitions $(s, \sigma, g^1, \lambda^1, s^1) \in \Delta$ and $(s, \sigma, g^2, \lambda^2, s^2) \in \Delta$, $g^1 \wedge g^2$ is not satisfiable. A *state* of $\mathcal{A}$ is a pair (s, v) of a location $s \in S$ and a valuation v for X. A *run* of $\mathcal{A}$ on a timed word $(\sigma_1, \tau_1)(\sigma_2, \tau_2) \cdots \in T\Sigma^\omega$ is a sequence of states $(s_0, v_0)(s_1, v_1)\dots$ where (i) $v_0 = \mathbf{0}$ and (ii) for each $i \geq 0$, there is a transition $(s_i, \sigma_{i+1}, g, \lambda, s_{i+1})$ such that $v_i + (\tau_{i+1} - \tau_i) \models g$ (let $\tau_0 = 0$) and $v_{i+1} = (v_i + (\tau_{i+1} - \tau_i))[\lambda \leftarrow 0]$. A run of $\mathcal{A}$ is *accepting* iff the set of locations it visits infinitely often contains at least one location from each F_i, $1 \leq i \leq n$. A timed word is *accepted* by $\mathcal{A}$ iff $\mathcal{A}$ has an accepting run on it. We denote by $[\![\mathcal{A}]\!]$ the timed language accepted by $\mathcal{A}$. For two TAs $\mathcal{A}^1 = \langle \Sigma, S^1, s_0^1, X^1, \Delta^1, \mathcal{F}^1 \rangle$ and $\mathcal{A}^2 = \langle \Sigma, S^2, s_0^2, X^2, \Delta^2, \mathcal{F}^2 \rangle$ over a common alphabet Σ, the (synchronous) product $\mathcal{A}^1 \times \mathcal{A}^2$ is defined as the TA $\langle \Sigma, S, s_0, X, \Delta, \mathcal{F} \rangle$ where (i) $S = S^1 \times S^2$, $s_0 = (s_0^1, s_0^2)$, and $X = X_1 \cup X_2$; (ii) $((s_1^1, s_1^2), \sigma, g, \lambda, (s_2^1, s_2^2)) \in \Delta$ iff there exists $(s_1^1, \sigma, g^1, \lambda^1, s_2^1) \in \Delta^1$ and $(s_1^2, \sigma, g^2, \lambda^2, s_2^2) \in \Delta^2$ such that $g = g^1 \wedge g^2$ and $\lambda = \lambda^1 \cup \lambda^2$; and (iii) let $\mathcal{F}^1 = \{F_1^1, \dots, F_n^1\}$, $\mathcal{F}^2 = \{F_1^2, \dots, F_m^2\}$, then $\mathcal{F} = \{F_1^1 \times S^2, \dots, F_n^1 \times S^2, S^1 \times F_1^2, \dots, S^1 \times F_m^2\}$. Note in particular that we have $[\![\mathcal{A}^1 \times \mathcal{A}^2]\!] = [\![\mathcal{A}^1]\!] \cap [\![\mathcal{A}^2]\!]$.

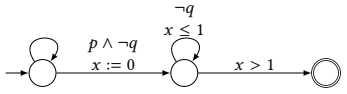

Fig. 3: A TA accepting $[\![\neg\, \mathbf{G}(p \Rightarrow \mathbf{F}_{\leq 1}\, q)]\!]$.

Example 2.1. Consider the TA over $\Sigma_{\{p,q\}}$ in Fig. 3 (following the usual convention, we omit transition labels when they are $\top$'s and use Boolean formulae over atomic propositions to represent letters, e.g., here $p \wedge \neg q$ stands for $\{\sigma \in \Sigma_{\{p,q\}} \mid p \in \sigma, q \notin \sigma\}$). It non-deterministically pick an event where p holds but q does not hold (thus $\mathbf{F}_{\leq}\, q$ is not fulfilled immediately) and enforces that q does not hold in the next time unit. In other words, it accepts $[\![\mathbf{F}\,(p \wedge \mathbf{G}_{\leq 1}(\neg q))]\!] = [\![\neg\, \mathbf{G}(p \Rightarrow \mathbf{F}_{\leq 1}\, q)]\!]$.

Alternation. One-clock alternating timed automata (OCATAs) extend one-clock timed automata with the power of *universal choice*. Intuitively, a transition of an OCATA may spawn several copies of the automaton that run in parallel from the targets of the transition; a timed word is accepted iff *all* copies accept it. Formally, for a set S of locations, let $\Gamma(S)$ be the set of formulae defined by

$$\gamma := \top \mid \bot \mid \gamma_1 \vee \gamma_2 \mid \gamma_1 \wedge \gamma_2 \mid s \mid x \bowtie c \mid x.\gamma$$

where x is the single clock, $c \in \mathbb{N}_{\geq 0}$, $\bowtie \in \{\leq, <, \geq, >\}$, and $s \in S$ (the construct $x.$ means "reset x"). For a formula $\gamma \in \Gamma(S)$, let its dual $\overline{\gamma} \in \Gamma(S)$ be the formula obtained by applying

- $\overline{\top} = \bot$; $\overline{\bot} = \top$;
- $\overline{\gamma_1 \vee \gamma_2} = \overline{\gamma_1} \wedge \overline{\gamma_2}$; $\overline{\gamma_1 \wedge \gamma_2} = \overline{\gamma_1} \vee \overline{\gamma_2}$;
- $\overline{s} = s$; $\overline{x \bowtie c} = \neg(x \bowtie c)$; $\overline{x.\gamma} = x.\overline{\gamma}$.

An OCATA over Σ is a tuple $\mathcal{A} = \langle \Sigma, S, s_0, \delta, F \rangle$ where S is a finite set of locations, $s_0 \in S$ is the initial location, $\delta \colon S \times \Sigma \to \Gamma(S)$ is the transition function, and $F \subseteq S$ is the set of final locations. A *state* of $\mathcal{A}$ is a pair (s, v) of a location $s \in S$ and a valuation v for the single clock x. Given a set of states M, a formula $\gamma \in \Gamma(S)$ and a clock valuation v, we define

- $M \models_v \top$; $M \models_v \ell$ iff $(\ell, v) \in M$; $M \models_v x \bowtie c$ iff $v \bowtie c$; $M \models_v x.\gamma$ iff $M \models_0 \gamma$;
- $M \models_v \gamma_1 \wedge \gamma_2$ iff $M \models_v \gamma_1$ and $M \models_v \gamma_2$;
- $M \models_v \gamma_1 \vee \gamma_2$ iff $M \models_v \gamma_1$ or $M \models_v \gamma_2$.

[7] Note that it is possible for $(\rho, i) \models \mathcal{A}_I(\varphi_1, \dots, \varphi_n)$ and $(\rho, i) \models \mathcal{A}_I^c(\varphi_1, \dots, \varphi_n)$, where $\mathcal{A}^c$ is the complement of $\mathcal{A}$, to hold simultaneously.

[8] We adopt *generalised Büchi* acceptance for technical convenience; indeed, any TA with a generalised Büchi acceptance condition can be converted into a classical Büchi TA via a simple standard construction [19].

We say that M is a *model* of γ with respect to v iff $M \models_v \gamma$.[9] A run of $\mathcal{A}$ on a timed word $(\sigma_1, \tau_1)(\sigma_2, \tau_2)\cdots \in T\Sigma^\omega$ is a rooted directed acyclic graph (DAG) $G = \langle V, \rightarrow\rangle$ with vertices of the form $(s, v, i) \in S \times \mathbb{R}_{\geq 0} \times \mathbb{N}_{\geq 0}$, $(s_0, 0, 0)$ as the root, and edges as follows: for every vertex (s, v, i), there is a model M of the formula $\delta(s, \sigma_{i+1})$ with respect to $v + (\tau_{i+1} - \tau_i)$ (again, $\tau_0 = 0$) such that there is an edge $(s, v, i) \rightarrow (s', v', i+1)$ for every state (s', v') in M. A run G of $\mathcal{A}$ is *accepting* iff every infinite path in G visits F infinitely often. A timed word is *accepted* by $\mathcal{A}$ iff $\mathcal{A}$ has an accepting run on it. We denote by $[\![\mathcal{A}]\!]$ the timed language accepted by $\mathcal{A}$. For convenience, in the sequel we will regard NFAs as (untimed) OCATAs with finite acceptance conditions and whose transition functions are simply disjunctions over locations.

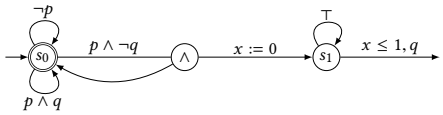

Fig. 4: An OCATA accepting $[\![\mathrm{G}(p \Rightarrow \mathrm{F}_{\leq 1}\, q)]\!]$.

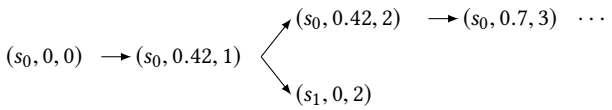

Fig. 5: A run of the OCATA in Fig. 4 on the timed word $(\emptyset, 0.42)(\{p\}, 0.42)(\{q\}, 0.7)\cdots$.

Example 2.2. Consider the OCATA over $\Sigma_{\{p,q\}}$ in Fig. 4 which accepts $[\![\mathrm{G}(p \Rightarrow \mathrm{F}_{\leq 1}\, q)]\!]$. A run of it on $(\emptyset, 0.42)(\{p\}, 0.42)(\{q\}, 0.7)\cdots$ is depicted in Fig. 5 where the root is $(s_0, 0, 0)$. This vertex has a single successor $(s_0, 0.42, 1)$, which in turn has two successors $(s_0, 0.42, 2)$ and $(s_1, 0, 2)$ (after firing the transition $\delta(s_0, \{p\}) = s_0 \wedge x.s_1$). Then, $(s_1, 0, 2)$ has no successor since the empty set is a model of $\delta(s_1, \{q\}) = x \leq 1$ with respect to 0.28.

Verification problems. In this work we are concerned with the following standard verification problems. Given an EMITL formula φ, the *satisfiability* problem asks whether $[\![\varphi]\!] = \emptyset$. Given a TA $\mathcal{A}$ and an EMITL formula φ, the *model-checking* problem asks whether $[\![\mathcal{A}]\!] \subseteq [\![\varphi]\!]$. As TAs are closed under intersection and the *emptiness* problem for TAs is decidable, both problems above can be solved by first translating φ into an equivalent TA $\mathcal{A}_\varphi$.

3 EXPRESSIVENESS

In this section we study the expressiveness of $\mathsf{EMITL}_{0,\infty}$, EECL, and a 'counting' extension of EMITL. It turned out that the class of timed languages captured by EMITL is robust in the sense that it remains the same under all these modifications. For the purpose of the proofs below, let us assume (without loss of generality) that the automaton $\mathcal{A} = \langle \Sigma, S, s_0, \delta, F\rangle$ in question is a DFA and at most one of $\varphi_1, \ldots, \varphi_n$ may hold at any position in a given timed word [50].

[9] Note that $\models_v$ is *monotonic*: if $M \subseteq M'$ and $M \models_v \gamma$ then $M' \models_v \gamma$.

Counting in intervals. Recall that the constraining intervals I in the counting modalities in Ex. 1.1 satisfy $0 \in I$; this non-trivial extension of MTL (and MITL) was first considered by Hirshfeld and Rabinovich [27, 28]. For the case of timed words, it is shown in [33] that allowing arbitrary I (e.g., $(1, 2)$) makes the resulting logic even more expressive. Here we show that, by contrast, adding the ability to count in I—regardless of whether $0 \in I$—does not increase the expressive power of EMITL.[10] We consider an extention of EMITL (which we call CEMITL) that enables specifying the number of positions within a given interval I from now at which final locations can be reached. More precisely, we have the following semantic clause in CEMITL:

- $(\rho, i) \models \mathcal{A}_I^{\geq k}(\varphi_1, \ldots, \varphi_n)$ iff there exists $j_1 < \cdots < j_k$ such that for each ℓ, $1 \leq \ell \leq k$, (i) $j_\ell \geq i$; (ii) $\tau_{j_\ell} - \tau_i \in I$; and (iii) there is an accepting run of $\mathcal{A}$ on some $a_i \ldots a_{j_\ell}$ where $a_{\ell'} \in \{1, \ldots, n\}$ and $(\rho, \ell') \models \varphi_{a_{\ell'}}$ for each ℓ', $i \leq \ell' \leq j_\ell$.

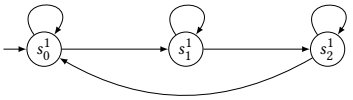

Fig. 6: $\mathcal{A}^1$ in the proof of Theorem 3.1.

Theorem 3.1. CEMITL *and* EMITL *are equally expressive over timed words.*

Proof. We give an EMITL equivalent of $\mathcal{A}_I^{\geq k}(\varphi_1, \ldots, \varphi_n)$. Provided that $\varphi_1, \ldots, \varphi_n$ are already in EMITL and $\mathcal{A}$ is deterministic in the sense above, we can count modulo k the number of positions where final locations are reached and ensures that I encompasses all possible values of the counter; in contrast to [33], here the counter can be implemented directly using automata modalities. We give a concrete example which should illustrate the idea. Let $k = 3$ and $\mathcal{A}^2$ be the product of $\mathcal{A}$ and $\mathcal{A}^1$ (Fig. 6), i.e. each location of $\mathcal{A}^2$ is of the form $\langle s, s^1\rangle$ where $s \in S$ and $s^1 \in \{s_0^1, s_1^1, s_2^1\}$, and it is accepting iff s and s^1 are both final. Then, let $\mathcal{A}^3$ be the automaton obtained from $\mathcal{A}^2$ by:

- For all the transitions $\langle s, s_0^1\rangle \rightarrow \langle s', s_1^1\rangle$, $\langle s, s_1^1\rangle \rightarrow \langle s', s_2^1\rangle$, and $\langle s, s_2^1\rangle \rightarrow \langle s', s_0^1\rangle$, keeping only those with $s' \in F$;
- For all the transitions $\langle s, s_0^1\rangle \rightarrow \langle s', s_0^1\rangle$, $\langle s, s_1^1\rangle \rightarrow \langle s', s_1^1\rangle$, and $\langle s, s_2^1\rangle \rightarrow \langle s', s_2^1\rangle$, keeping only those with $s' \notin F$.

Now let $\mathcal{A}^{1,\ell}$ ($\ell \in \{0, 1, 2\}$) be the automaton obtained from $\mathcal{A}^1$ by adding an extra final location s_F^1 and the transition $s^1_{\ell-1 \pmod 3} \rightarrow s_F^1$, and let $\mathcal{A}^{3,\ell}$ be the corresponding product with $\mathcal{A}$, keeping transitions $\langle s, s^1_{\ell-1 \pmod 3}\rangle \rightarrow \langle s', s_F^1\rangle$ with $s' \in F$. The original formula $\mathcal{A}_I^{\geq 3}$ is equivalent to $\bigwedge_{\ell \in \{0,1,2\}} \mathcal{A}_I^{3,\ell}$. □

Restricting to event clocks. We show that the equivalence of ECL and $\mathsf{MITL}_{0,\infty}$ carries over to the current setting. More specifically, an EECL formula can be translated into an equilvalent $\mathsf{EMITL}_{0,\infty}$ formula of polynomial size (in DAG representation). On the other

[10] As EMITL can easily express the 'until with threshold' modalities of CTMITL, the latter is clearly subsumed by EMITL.

hand, our translation from $\mathsf{MITL}_{0,\infty}$ to EECL induces an exponential blow-up due to the fact that automata $\mathcal{A}$ have to be determinised.

Theorem 3.2. EECL *and* $\mathsf{EMITL}_{0,\infty}$ *are equally expressive over timed words.*

Proof. Again, we assume that the arguments $\varphi_1, \ldots, \varphi_n$ are already in the target logic. The direction from EECL to $\mathsf{EMITL}_{0,\infty}$ is simple and almost identical to the translation from ECL to $\mathsf{MITL}_{0,\infty}$; for example, $\overset{\triangleright}{\mathcal{A}}_{(3,5)}$ can be written as $\mathcal{A}_{<5} \wedge \neg\mathcal{A}_{\leq 3}$. For the other direction consider the following $\mathsf{EMITL}_{0,\infty}$ formulae:

- $(\rho, i) \models \mathcal{A}_{\leq c}$: the equivalent formula is simply $\overset{\triangleright}{\mathcal{A}}_{\leq c}$.
- $(\rho, i) \models \mathcal{A}_{\geq c}$: as in [43], we consider the subcases where:
  - There is no event in $[\tau_i, \tau_i + c)$ apart from (σ_i, τ_i): let $\mathcal{A}^2$ be the product of $\mathcal{A}$ and $\mathcal{A}^1$ where $\mathcal{A}^1$ is the automaton depicted in Fig. 7. We have $(\rho, i) \models \neg\, \overset{\triangleright}{\mathcal{A}}^1_{<c} \wedge \overset{\triangleright}{\mathcal{A}}^2_{\geq c}$.
  - There are events in $[\tau_i, \tau_i + c)$ other than (σ_i, τ_i): let the last event in $[\tau_i, \tau_i + c)$ be (σ_j, τ_j) and $k > j > i$ be the minimal position such that there exists $a_i \ldots a_k \in [\![\mathcal{A}]\!]$ with $(\rho, \ell) \models \varphi_{a_\ell}$ for all ℓ, $i \leq \ell \leq k$. By assumption, $a_i \ldots a_k$ is unique and $\mathcal{A}$ must reach a specific location $s \in S$ after reading $a_i \ldots a_j$. The idea is to split the unique run of $\mathcal{A}$ on $a_i \ldots a_k$ at s: we take a disjunction over all possible $s \in S$, enforce that $\tau_j - \tau_i < c$ and $\mathcal{A}$ reaches a final location from s by reading $a_{j+1} \ldots a_k$. More specifically, let $\mathcal{B}^{s,\varphi}$ be the automaton obtained from $\mathcal{A}$ by adding a new location s_F, declaring it as the only final location, and adding new transitions $s' \xrightarrow{\varphi_a \wedge \varphi} s_F$ for every $s' \xrightarrow{\varphi_a} s$ in $\mathcal{A}$. Let $\mathcal{C}^s$ be the automaton obtained from $\mathcal{A}$ by adding new non-final locations s'_0 and s'_1, adding new transitions $s'_0 \to s'_1$ (i.e. labelled with $\top$) and $s'_1 \xrightarrow{\varphi_a} s''$ for every $s \xrightarrow{\varphi_a} s''$ in $\mathcal{A}$, removing outgoing transitions from all the final locations, and finally setting the initial location to s'_0. We have $(\rho, i) \models \overset{\triangleright}{\mathcal{A}}^1_{<c} \wedge \overset{\triangleright}{\mathcal{A}} \wedge \neg \bigvee_{s \in S} \overset{\triangleright}{\mathcal{B}}^{s,\varphi}_{<c}$ where $\varphi = \neg\mathcal{C}^s$.

  The equivalent formula is the disjunction of these.

The other types of constraing intervals, such as $[0, c)$, are handled almost identically. □

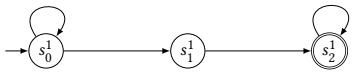

Fig. 7: $\mathcal{A}^1$ in the proof of Theorem 3.2.

Restricting to one-sided constraining intervals. Recall that a fundamental stumbling block in the algorithmic analysis of TAs is that the *universality* problem is undecidable [2]. DTAs with finite acceptance conditions, on the other hand, can be complemented easily and have a decidable universality problem. This raises the question of whether one can extend MITL with DTA modalities without losing decidability (both are fully decidable formalisms). Perhaps surprisingly, the resulting formalism already subsumes MTL even when punctual constraints are disallowed. For example, $\mathbf{F}_{[d,d]}\,\varphi$ can be written as $\neg\mathcal{A}' \wedge \neg\mathcal{A}'' \wedge \mathbf{F}_{[1,\infty)}\,\varphi$ where $\mathcal{A}'$ and $\mathcal{A}''$ are the one-clock deterministic TAs in Fig. 8 and Fig. 9, respectively (in particular, note that $\mathcal{A}'$ and $\mathcal{A}''$ only use lower- and upper-bound constraints). It follows from [40] that the satisfiability problem for this formalism is undecidable. Based on a

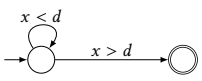

Fig. 8: $\mathcal{A}'$.

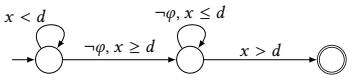

Fig. 9: $\mathcal{A}''$.

similar trick, we obtain the main result of this section: $\mathsf{EMITL}_{0,\infty}$ already has the full expressive power of EMITL. This, together with the fact that the satisfiability and model-checking problems for $\mathsf{EMITL}_{0,\infty}$ are only PSPACE-complete (Theorem 4.6) as compared with EXPSPACE-complete for full EMITL [21], makes $\mathsf{EMITL}_{0,\infty}$ a competitive alternative to other real-time specification formalisms—while a translation from EMITL to $\mathsf{EMITL}_{0,\infty}$ inevitably induces at least an exponential blow-up, it can be argued that many properties of practical interest can be written in $\mathsf{EMITL}_{0,\infty}$ directly (e.g., Ex. 1.1 and Ex. 1.2). The idea of the proof below is similar to that of [43, Lemma 6.3.11] ($\mathsf{MITL}_{0,\infty}$ and MITL are equally expressive in the continuous semantics), but the technical details are more involved due to automata modalities and the fact that each event is not necessarily preceded by another one exactly 1 time unit earlier in a timed word; the latter is essentially the reason why the expressive equivalence of $\mathsf{MITL}_{0,\infty}$ and MITL fails to hold in the pointwise semantics.

Theorem 3.3. $\mathsf{EMITL}_{0,\infty}$ *and* EMITL *are equally expressive over timed words.*

Proof. We explain in detail below how to write the EMITL formula $\mathcal{A}_{(c,c+1)}(\varphi_1, \ldots, \varphi_n)$ where $c \geq 0$, and $\varphi_1, \ldots, \varphi_n \in \mathsf{EMITL}_{0,\infty}$ as an $\mathsf{EMITL}_{0,\infty}$ formula; the other cases, such as $(c, c+1]$ and $[c, c+1]$, are similar.

First consider $c = 0$. If $(\rho, i) \models \mathcal{A}_{(0,1)}$ for $\rho = (\sigma_1, \tau_1)(\sigma_2, \tau_2)\ldots$ and $i \geq 1$, the finite word $a_i \ldots a_k$ accepted by $\mathcal{A}$ must be at least two letters long. This again is enforced by $\mathcal{A}^1$ in Fig. 7: let $\mathcal{A}^2$ be the product of $\mathcal{A}$ and $\mathcal{A}^1$. Then, let $\mathcal{A}^3$ be the automaton obtained from $\mathcal{A}^2$ by adding $\neg\,\mathbf{X}_{>0}\,\top$ ($\mathbf{X}$ is the standard MITL 'next' operator [41]) to all the transitions $\langle s, s^1_0\rangle \to \langle s', s^1_0\rangle$ and $\mathbf{X}_{>0}\,\top$ to all the transitions $\langle s, s^1_0\rangle \to \langle s', s^1_1\rangle$ as conjuncts (in doing so, extend the alphabet as necessary). It is not hard to see that $(\rho, i) \models \mathcal{A}^3_{<1}$ in the two possible situations: (i) $\tau_{i+1} - \tau_i > 0$ and (ii) $\tau_j - \tau_i > 0$ for some $j > i+1$ and $\tau_\ell - \tau_i = 0$ for all ℓ, $i < \ell < j$. The other direction ($(\rho, i) \models \mathcal{A}^3_{<1} \Rightarrow (\rho, i) \models \mathcal{A}_{(0,1)}$) is straightforward. It follows that the equivalent $\mathsf{EMITL}_{0,\infty}$ formula is $\mathcal{A}^3_{<1}$.

Now consider $c > 0$. Suppose that $(\rho, i) \models \mathcal{A}_{(c,c+1)}$ for $\rho = (\sigma_1, \tau_1)(\sigma_2, \tau_2)\ldots$ and $i \geq 1$, let $k > i$ be the *minimal* position such that $\tau_k - \tau_i \in (c, c+1)$ and there exists $a_i \ldots a_k \in [\![\mathcal{A}]\!]$ with $(\rho, \ell) \models \varphi_{a_\ell}$ for all ℓ, $i \leq \ell \leq k$ (since at most one of $\varphi_1, \ldots, \varphi_n$ may hold at any position, we fix $a_i \ldots a_k$ below). Consider the following cases (note that they are not mutually disjoint):

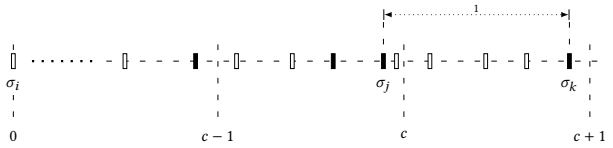

Fig. 10: Case (i) in the proof of Theorem 3.3; solid boxes indicate when $\mathcal{A}$ accepts the corresponding prefix of $a_i \dots a_k$.

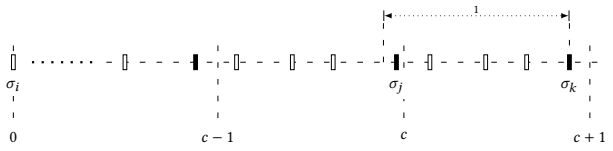

Fig. 11: Case (ii) in the proof of Theorem 3.3.

(i) There exists a *maximal* j, $i < j < k$ such that (a) $\tau_k - \tau_j = 1$ and (b) there is no ℓ, $j < \ell < k$ such that $a_i \dots a_\ell \in [\![\mathcal{A}]\!]$ with $(\rho, \ell') \models \varphi_{a_{\ell'}}$ for all ℓ', $i \le \ell' \le \ell$ (Fig. 10): we take a disjunction over all possible $s \in S$ such that $\mathcal{A}$ reaches s after reading $a_i \dots a_j$ and enforce that $\tau_j - \tau_i \in (c-1, c)$ (which we can, by the IH), $\mathcal{A}$ reaches a final location from s by reading $a_{j+1} \dots a_k$, and $\tau_k - \tau_j = 1$; thanks to (b), the last condition, which is otherwise inexpressible in $\mathsf{EMITL}_{0,\infty}$, can be expressed as a conjunction of two formulae labelled with ≤ 1 and ≥ 1. To this end, we use $\mathcal{B}^{s,\varphi}$ and C^s as defined in the proof of Theorem 3.2: we have

$$(\rho, i) \models \varphi^1 = \bigvee_{s \in S} \mathcal{B}^{s,\varphi}_{(c-1,c)}$$

where $\varphi = C^s_{\le 1} \wedge C^s_{\ge 1}$.

(ii) There exists j, $i < j < k$ such that $\tau_k - \tau_j < 1$, $\tau_j - \tau_i \in (c-1, c]$, and $a_i \dots a_j \in [\![\mathcal{A}]\!]$ with $(\rho, \ell) \models \varphi_{a_\ell}$ for all ℓ, $i \le \ell \le j$ (Fig. 11): let $\mathcal{D}^s$ be the automaton obtained from $\mathcal{A}$ in the same way as C^s except that we do not remove outgoing transitions from the final locations. Regardless of whether there is a event at $\tau_k - 1$, it is clear that every position ℓ with $\tau_\ell - \tau_i \in (c-1, c]$ must satisfy $C^s_{(0,1)}$ where s is the location of $\mathcal{A}$ after reading $a_i \dots a_\ell$. We have

$$(\rho, i) \models \varphi^2 = \mathcal{A}_{(c-1,c]} \wedge \neg \bigvee_{s \in S} \mathcal{B}^{s,\varphi}_{(c-1,c]}$$

where $\varphi = \neg \mathcal{D}^s_{(0,1)}$.

(iii) There exists j, $i < j < k$ such that $\tau_k - \tau_j < 1$, $\tau_j - \tau_i \in (c-1, c]$, but there is no ℓ, $i < \ell < k$ such that (a) $\tau_\ell - \tau_i \in (c-1, c]$ and (b) $a_i \dots a_\ell \in [\![\mathcal{A}]\!]$ with $(\rho, \ell') \models \varphi_{a_{\ell'}}$ for all ℓ', $i \le \ell' \le \ell$: we have

$$(\rho, i) \models \varphi^3 = \neg \mathcal{A}_{(c-1,c]} \wedge \bigvee_{s \in S} \mathcal{B}^{s,\varphi}_{(c-1,c]}$$

where $\varphi = \mathcal{D}^s_{(0,1)}$.

(iv) There exists a *maximal* j, $i < j < k$ such that $\tau_k - \tau_j > 1$, $\tau_j - \tau_i \in (c-1, c]$, and there is no ℓ, $j < \ell < k$ such that (a) $\tau_\ell - \tau_i \in (c-1, c]$ and (b) $a_i \dots a_\ell \in [\![\mathcal{A}]\!]$ with $(\rho, \ell') \models \varphi_{a_{\ell'}}$ for all ℓ', $i \le \ell' \le \ell$: observe that (provided that s's are correctly instantiated to the locations $\mathcal{A}$ reaches as it reads $a_i \dots a_k$) while $C^s_{>1}$ may hold arbitrarily often in $[\tau_i, \tau_i + c]$, the number of positions ℓ, $i \le \ell \le j$ satisfying

$$(\rho, \ell) \models C^s_{>1} \wedge (\rho, \ell + 1) \models \left(C^s_{\le 1} \vee (C^s_{>1} \wedge s \in F)\right) \tag{2}$$

is at most c (since any two of such positions must be separated by more than 1 time unit). We define a family of automata modalities $\{\mathcal{E}^m \mid m \ge 1\}$ such that each location of $\mathcal{E}^m$ is of the form $\langle s, d \rangle$ with $s \in S$ and $1 \le d \le 2m$; see Fig. 12 for an illustration. Each transition updates the s-component as $\mathcal{A}$ would, enforces the formula labelled on the corresponding transition of $\mathcal{A}$ and, additionally, the formula as labelled in Fig. 12 (with s being the *target* location of the corresponding transition of $\mathcal{A}$). The formula φ^s (illustrated in Fig. 13), which also follows $\mathcal{A}$ with an s-component, checks that the next position either satisfies (a) $s \in F$, or (b) $C^s_{\le 1}$ holds continuously until $s \in F$ eventually holds. Let $\hat{\mathcal{E}}^m$ be obtained from $\mathcal{E}^m$ by 'inlining' φ^s: removing the leftmost locations of φ^s and merge the middle locations of φ^s with the rightmost locations of $\mathcal{E}^m$. Apparently, $\hat{\mathcal{E}}^m$ and $\mathcal{E}^m$ are equivalent if there is no constraing interval—the only difference between them is which position is 'timed'. Now suppose that the number of positions ℓ, $i \le \ell \le j$ satisfying (2) is m. Since j is the last of these positions, we have $(\rho, i) \models \hat{\mathcal{E}}^m_{<c+1}$. On the other hand, as there are only $m-1$ such positions in $[\tau_i, \tau_i + c - 1]$, we have $(\rho, i) \models \neg \mathcal{E}^m{}_{\le c-1}$. By the above, we have

$$(\rho, i) \models \varphi^4 = \bigvee_{1 \le m \le c} \left(\hat{\mathcal{E}}^m_{<c+1} \wedge \neg \mathcal{E}^m{}_{\le c-1}\right).$$

(v) There is no event in $(\tau_i + c - 1, \tau_i + c]$: We have

$$(\rho, i) \models \varphi^5 = \neg\, \mathbf{F}_{(c-1,c]}\, \top \wedge \varphi'.$$

If $c = 1$ then φ' can simply be taken as $\mathcal{A}_{(0,2)}$, which is equivalent to $\mathcal{A}^3_{<2}$ by the same argument as before. If $c > 1$, then φ' can be taken as

$$\bigvee_{1 \le m \le c-1} \left(\hat{\mathcal{E}}^m_{<c+1} \wedge \neg \mathcal{E}^m{}_{\le c-2}\right) \vee \varphi''$$

where φ'' is

$$\neg\, \mathbf{F}_{(c-2,c-1]}\, \top \wedge \bigvee_{1 \le m \le c-2} \left(\hat{\mathcal{E}}^m_{<c+1} \wedge \neg \mathcal{E}^m{}_{\le c-3}\right) \vee \varphi'''.$$

Intuitively, the former part of φ' is used to handle the case when there is (at least) a event in $(\tau_i + c - 2, \tau_i + c - 1]$, and

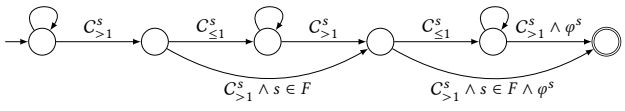

Fig. 12: An illustration of $\mathcal{E}^3$ in the proof of Theorem 3.3.

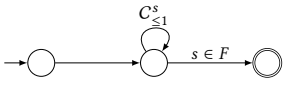

Fig. 13: An illustration of φ^s in the proof of Theorem 3.3.

the former part of φ'' is for $(\tau_i + c - 3, \tau_i + c - 2]$, and so on.

We omit the other direction as it is (more or less) straightforward. The equivalent $\mathsf{EMITL}_{0,\infty}$ formula is $\varphi^1 \vee \varphi^2 \vee \varphi^3 \vee \varphi^4 \vee \varphi^5$. □

4 FROM $\mathsf{EMITL}_{0,\infty}$ TO TIMED AUTOMATA

Embedding EMITL *formulae into* OCATA*s.* We give a translation from a given EMITL formula φ over AP (which we assume to be in negative normal form) into an OCATA $\mathcal{A}_\varphi = \langle \Sigma_{\mathsf{AP}}, S, s_0, \delta, F \rangle$ such that $[\![\mathcal{A}_\varphi]\!] = [\![\varphi]\!]$. While this mostly follows the lines of the translation for MTL (and MITL) in [16, 41], it is worth noting that the resulting OCATA $\mathcal{A}_\varphi$ is *weak* [36, 37] but not necessarily *very-weak* [24, 44] due to the presence of automata modalities. The set of locations S of $\mathcal{A}_\varphi$ contains (i) s^{init}; (ii) all the locations of $\mathcal{A}$ for every subformula $\mathcal{A}_I(\varphi_1, \ldots, \varphi_n)$; (iii) all the locations of $\mathcal{A}$ for every subformula $\tilde{\mathcal{A}}_I(\varphi_1, \ldots, \varphi_n)$. The initial location s_0 is s^{init}, and the final locations F are all the locations of $\mathcal{A}$ for every subformula $\tilde{\mathcal{A}}_I(\varphi_1, \ldots, \varphi_n)$. Finally, for each $\sigma \in \Sigma_{\mathsf{AP}}$, δ is defined inductively as follows (let $\mathcal{A} = \langle \Sigma^{\mathcal{A}}, S^{\mathcal{A}}, s_0^{\mathcal{A}}, \delta^{\mathcal{A}}, F^{\mathcal{A}} \rangle$ with $\Sigma^{\mathcal{A}} = \{1, \ldots, n\}$):

- $\delta(s^{init}, \sigma) = x.\delta(\varphi, \sigma)$, $\delta(\top, \sigma) = \top$, and $\delta(\bot, \sigma) = \bot$;
- $\delta(p, \sigma) = \top$ if $p \in \sigma$, $\delta(p, \sigma) = \bot$ otherwise;
- $\delta(\neg p, \sigma) = \top$ if $p \notin \sigma$, $\delta(\neg p, \sigma) = \bot$ otherwise;
- $\delta(\varphi_1 \vee \varphi_2, \sigma) = \delta(\varphi_1, \sigma) \vee \delta(\varphi_2, \sigma)$, and $\delta(\varphi_1 \wedge \varphi_2, \sigma) = \delta(\varphi_1, \sigma) \wedge \delta(\varphi_2, \sigma)$;
- $\delta(\mathcal{A}_I(\varphi_1, \ldots, \varphi_n), \sigma) = x.\delta(s_0^{\mathcal{A}}, \sigma)$;
- $\delta(s^{\mathcal{A}}, \sigma) = \bigvee_{a \in \Sigma^{\mathcal{A}}} \left(\delta(\varphi_a, \sigma) \wedge \delta^{\mathcal{A}}[s_F^{\mathcal{A}} \leftarrow s_F^{\mathcal{A}} \vee x \in I](s^{\mathcal{A}}, a)\right)$ where $s^{\mathcal{A}} \in S^{\mathcal{A}}$ and $\delta^{\mathcal{A}}[s_F^{\mathcal{A}} \leftarrow s_F^{\mathcal{A}} \vee x \in I]$ is obtained from $\delta^{\mathcal{A}}$ by substituting every $s_F^{\mathcal{A}} \in F^{\mathcal{A}}$ with $s_F^{\mathcal{A}} \vee x \in I$ for some subformula $\mathcal{A}_I(\varphi_1, \ldots, \varphi_n)$;
- $\delta(\tilde{\mathcal{A}}_I(\varphi_1, \ldots, \varphi_n), \sigma) = x.\delta(s_0^{\mathcal{A}}, \sigma)$;
- $\delta(s^{\mathcal{A}}, \sigma) = \bigwedge_{a \in \Sigma^{\mathcal{A}}} \left(\delta(\varphi_a, \sigma) \vee \overline{\delta^{\mathcal{A}}}[s_F^{\mathcal{A}} \leftarrow s_F^{\mathcal{A}} \wedge x \notin I](s^{\mathcal{A}}, a)\right)$ where $s^{\mathcal{A}} \in S^{\mathcal{A}}$ and $\overline{\delta^{\mathcal{A}}}[s_F^{\mathcal{A}} \leftarrow s_F^{\mathcal{A}} \wedge x \notin I]$ is obtained from $\overline{\delta^{\mathcal{A}}}$ by substituting every $s_F^{\mathcal{A}} \in F^{\mathcal{A}}$ with $s_F^{\mathcal{A}} \wedge x \notin I$ for some subformula $\tilde{\mathcal{A}}_I(\varphi_1, \ldots, \varphi_n)$.

Proposition 4.1. *Given an* EMITL *formula φ in negative normal form, $[\![\mathcal{A}_\varphi]\!] = [\![\varphi]\!]$.*

We now focus on the case where φ is an $\mathsf{EMITL}_{0,\infty}$ formula and give a set of component TAs whose product 'implements' the corresponding OCATA $\mathcal{A}_\varphi$. As we will need some notions from [17], we briefly recall them here to keep the paper self-contained.

Compositional removal of alternation in φ. Let Φ be the set of temporal subformulae (i.e. whose outermost operator is $\mathcal{A}_I$ or $\tilde{A}_I$) of φ. We introduce a new atomic proposition p_ψ for each $\psi \in \Phi$ (the *trigger* for ψ) and let $\mathsf{AP}_\Phi = \{p_\psi \mid \psi \in \Phi\}$. For a timed word ρ' over $\Sigma_{\mathsf{AP} \cup \mathsf{AP}_\Phi}$, we denote by $\mathrm{proj}_{\mathsf{AP}}(\rho')$ the timed word obtained from ρ' by hiding all $p \notin \mathsf{AP}$ (i.e. $p \in \mathsf{AP}_\Phi$). For a timed language $\mathcal{L}$ over $\mathsf{AP} \cup \mathsf{AP}_\Phi$ we write $\mathrm{proj}_{\mathsf{AP}}(\mathcal{L}) = \{\mathrm{proj}_{\mathsf{AP}}(\rho') \mid \rho' \in \mathcal{L}\}$. Let $\overline{\psi}$ be the formula obtained from an $\mathsf{EMITL}_{0,\infty}$ formula ψ (in negative normal form) by replacing all of its top-level temporal subformulae by their corresponding triggers, i.e. $\overline{\psi}$ is defined inductively as follows (where $p \in \mathsf{AP}$):

- $\overline{\psi_1 \wedge \psi_2} = \overline{\psi_1} \wedge \overline{\psi_2}$;
- $\overline{\psi_1 \vee \psi_2} = \overline{\psi_1} \vee \overline{\psi_2}$;
- $\overline{\psi} = \psi$ when ψ is $\top$ or $\bot$ or p or $\neg p$;
- $\overline{\psi} = p_\psi$ when ψ is $\mathcal{A}_I(\varphi_1, \ldots, \varphi_n)$ or $\tilde{\mathcal{A}}_I(\varphi_1, \ldots, \varphi_n)$.

Note that $\overline{\psi}$ is simply a positive Boolean combination of atomic propositions. In this way, we can turn the given $\mathsf{EMITL}_{0,\infty}$ formula φ into an equisatisfiable $\mathsf{EMITL}_{0,\infty}$ formula φ' over $\mathsf{AP} \cup \mathsf{AP}_\Phi$: the conjunction of $\overline{\varphi}$,

$$\bigwedge_{\{\psi \in \Phi \mid \psi = \mathcal{A}_I(\varphi_1, \ldots, \varphi_n)\}} \mathrm{G}\left(p_\psi \Rightarrow \mathcal{A}_I(\overline{\varphi_1}, \ldots, \overline{\varphi_n})\right),$$

and the counterparts for $\{\psi \in \Phi \mid \psi = \tilde{\mathcal{A}}_I(\varphi_1, \ldots, \varphi_n)\}$. Finally, we construct the component TAs C^{init} (which accepts $[\![\overline{\varphi}]\!]$) and C^ψ (which accepts, say, $[\![\mathrm{G}\left(p_\psi \Rightarrow \mathcal{A}_I(\overline{\varphi_1}, \ldots, \overline{\varphi_n})\right)]\!]$) for every $\psi \in \Phi$. The timed language of φ' is accepted by the product $C^{init} \times \prod_{\psi \in \Phi} C^\psi$ and, in particular, $\mathrm{proj}_{\mathsf{AP}}([\![C^{init} \times \prod_{\psi \in \Phi} C^\psi]\!]) = [\![\mathcal{A}_\varphi]\!]$. Intuitively, p_ψ being $\top$ (the trigger p_ψ is '*pulled*') at some position means that the OCATA $\mathcal{A}_\varphi$ spawns a copy (several copies) of $\mathcal{A}$ where $\psi = \mathcal{A}_I$ ($\psi = \tilde{\mathcal{A}}_I$) at this position or, equivalently, an obligation that $\mathcal{A}_I$ ($\tilde{\mathcal{A}}_I$) must hold is imposed on this position.

Component TA*s for automata modalities.* As the construction of C^{init} is trivial, we only describe the component TAs C^ψ for $\psi = \mathcal{A}_I(\varphi_1, \ldots, \varphi_n)$ and $\psi = \tilde{\mathcal{A}}_I(\varphi_1, \ldots, \varphi_n)$ where $I = [0, c]$ or $I = [c, \infty)$ for some $c \in \mathbb{N}_{\geq 0}$; the other types of constraining intervals are handled similarly. The crucial observation that allows us to bound the number of clocks needed in C^ψ is that two or more obligations, provided that their corresponding copies of $\mathcal{A}$ are in the same location(s) at some point(s) and I is a lower or upper bound, can be merged into a single one. Instead of keeping track of the order of the values of its clocks, C^ψ non-deterministically guesses how obligations should be merged and put them into suitable sub-components accordingly. To ensure that all obligations are satisfied, we use an extra variable ℓ such that $\ell = 0$ when there is no obligation, $\ell = 1$ when there is at least one pending obligation, $\ell = 2$ when the pending obligations have just been satisfied and a new obligation has just arrived, and finally $\ell = 3$ when we have to wait the current obligations to be satisfied (explained below). In all the cases below we fix $\Sigma = \Sigma_{\mathsf{AP} \cup \mathsf{AP}_\Phi}$ and $|S^{\mathcal{A}}| = m$. We write $S^{src} \xrightarrow[\vee]{\sigma} S^{tgt}$, where S^{src} and S^{tgt} are two subsets of $S^{\mathcal{A}}$, iff S^{tgt} is a minimal set such that for each $s^{\mathcal{A},1} \in S^{src}$, there is a transition $s^{\mathcal{A},1} \xrightarrow{\overline{\varphi_a}} s^{\mathcal{A},2}$ (where $a \in \{1, \ldots, n\}$) of $\mathcal{A}$ with $\sigma \models \overline{\varphi_a}$ and $s^{\mathcal{A},2} \in S^{tgt}$. Similarly, we write $S^{src} \xrightarrow[\wedge]{\sigma} S^{tgt}$ iff S^{tgt} is a minimal set such that for each $s^{\mathcal{A},1} \in S^{src}$ and each transition $s^{\mathcal{A},1} \xrightarrow{\overline{\varphi_a}} s^{\mathcal{A},2}$ (where $a \in \{1, \ldots, n\}$) of $\mathcal{A}$, either $s^{\mathcal{A},2} \in S^{tgt}$ or $\sigma \models \overline{\varphi_a}$.

$\psi = \mathcal{A}_{\leq c}$. Let $C^\psi = \langle \Sigma, S, s_0, X, \Delta, \mathcal{F} \rangle$ be defined as follows (to simplify the presentation, in this case we assume that $\mathcal{A}$ only accepts words of length ≥ 2):

- Each location $s \in S$ is of the form $\langle \ell_1, S_1, \ldots, \ell_m, S_m \rangle$ where $\ell_j \in \{0, 1, 2\}$ and $S_j \subseteq S^{\mathcal{A}}$ for all $j \in \{1, \ldots, m\}$; intuitively, $\langle \ell_j, S_j \rangle$ can be seen as a location of the sub-component C_j^ψ;
- $s_0 = \langle 0, \emptyset, \ldots, 0, \emptyset \rangle$;
- $X = \{x_1, \ldots, x_m\}$;

- $\mathcal{F} = \{F_1, \ldots, F_m\}$ where F_j contains all locations with $\ell_j = 0$ or $\ell_j = 2$;
- Δ is obtained by synchronising the transitions $\langle \ell, S\rangle \xrightarrow{\sigma, g, \lambda} \langle \ell', S'\rangle$ of individual sub-components (we omit the subscripts for brevity):
  - $p_\psi \notin \sigma$; $\ell' = 0$, $\ell = 0$; $S' = \emptyset$, $S = \emptyset$; $g = \top$; $\lambda = \emptyset$.
  - $p_\psi \notin \sigma$; $\ell' = 1$, $\ell \in \{1, 2\}$; $S \xrightarrow[\vee]{\sigma} S'$; $g = \top$; $\lambda = \emptyset$.
  - $p_\psi \notin \sigma$; $\ell' = 0$, $\ell \in \{1, 2\}$; $S' = \emptyset$, $S = \{s^{\mathcal{A}}\}$, $s_F^{\mathcal{A}} \models \delta(s^{\mathcal{A}}, \sigma)$ for some $s_F^{\mathcal{A}} \in F^{\mathcal{A}}$; $g = x \leq c$; $\lambda = \emptyset$.
  - $p_\psi \in \sigma$; $\ell' = 1$, $\ell = 0$; $\{s_0^{\mathcal{A}}\} \xrightarrow[\vee]{\sigma} S'$, $S = \emptyset$; $g = \top$; $\lambda = \{x\}$.
  - $p_\psi \in \sigma$; $\ell' = 1$, $\ell \in \{1, 2\}$; S' is the union of some S'' such that $S \xrightarrow[\vee]{\sigma} S''$ and S''' with $\{s_0^{\mathcal{A}}\} \xrightarrow[\vee]{\sigma} S'''$; $g = \top$; $\lambda = \emptyset$.
  - $p_\psi \in \sigma$; $\ell' = 2$, $\ell \in \{1, 2\}$; $\{s_0^{\mathcal{A}}\} \xrightarrow[\vee]{\sigma} S'$, $S = \{s^{\mathcal{A}}\}$, $s_F^{\mathcal{A}} \models \delta(s^{\mathcal{A}}, \sigma)$ for some $s_F^{\mathcal{A}} \in F^{\mathcal{A}}$; $g = x \leq c$; $\lambda = \{x\}$.

  If $p_\psi \in \sigma$, then exactly one of the sub-components takes a '$p_\psi \in \sigma$' transition while the others proceed as if $p_\psi \notin \sigma$.

Proposition 4.2. $[\![C^\psi]\!] = [\![\mathbf{G}\left(p_\psi \Rightarrow \mathcal{A}_{\leq c}(\overline{\varphi_1}, \ldots, \overline{\varphi_n})\right)]\!]$.

Proof sketch. Let $\psi' = \mathbf{G}\left(p_\psi \Rightarrow \mathcal{A}_{\leq c}(\overline{\varphi_1}, \ldots, \overline{\varphi_n})\right)$ and $\mathcal{A}_{\psi'}$ be the equivalent OCATA obtained via Proposition 4.1. If a timed word $\rho = (\sigma_i, \tau_i)_{i \geq 1}$ satisfies ψ', there must be an accepting run $G = \langle V, \rightarrow\rangle$ of $\mathcal{A}_{\psi'}$ on ρ; in particular, a copy of $\mathcal{A}$ is spawned whenever p_ψ holds. Now consider each 'level' $L_i = \{(s, v) \mid (s, v, i) \in V\}$ of G in the increasing order of i. If $|\{(s^{\mathcal{A}}, v) \mid (s^{\mathcal{A}}, v) \in L_i\}| \leq 1$ for every $s^{\mathcal{A}} \in S^{\mathcal{A}}$, in C^ψ we simply put each corresponding obligation into an unused sub-component (with $\ell = 0$ and $S = \emptyset$) when it arrives, i.e. p_ψ holds. If $|\{(s^{\mathcal{A}}, v) \mid (s^{\mathcal{A}}, v) \in L_i\}| > 1$ for some $s^{\mathcal{A}} \in S^{\mathcal{A}}$, since the constraining interval $[0, c]$ is *downward closed*, the DAG obtained from G by replacing all the subtrees rooted at nodes $(s^{\mathcal{A}}, v, i)$ with $(s^{\mathcal{A}}, v^{max}, i)$, where $v^{max} = \max\{v \mid (s^{\mathcal{A}}, v) \in L_i\}$, is still an accepting run of $\mathcal{A}_{\psi'}$; in C^ψ, this amounts to putting the obligations that correspond to nodes $(s^{\mathcal{A}}, v, i)$ into the sub-component that holds the (oldest) obligation that corresponds to $(s^{\mathcal{A}}, v^{max}, i)$. We do the same for all such $s^{\mathcal{A}}$, obtain G', and start over from $i + 1$. In this way, we can readily construct an accepting run of C^ψ on ρ. The other direction obviously holds as each sub-component C_j^ψ does not reset its associated clock x_j when $p_\psi \in \sigma$ and $\ell_j \in \{1, 2\}$, unless the (only remaining) obligation in S_j is fulfilled right away. In other words, C_j^ψ adds an obligation that is at least as strong to S_j without weakening the existing ones in S_j. □

$\psi = \mathcal{A}_{\geq c}$. Let $C^\psi = \langle \Sigma, S, s_0, X, \Delta, \mathcal{F}\rangle$ be defined as follows:

- Each location $s \in S$ is of the form $\langle \ell_1, S_1, T_1 \ldots, \ell_m, S_m, T_m\rangle$ where $\ell_j \in \{0, 1, 2, 3\}$ and $S_j, T_j \subseteq S^{\mathcal{A}}$ for all $j \in \{1, \ldots, m\}$; intuitively, $\langle \ell_j, S_j, T_j\rangle$ can be seen as a location of the sub-component C_j^ψ;
- $s_0 = \langle 0, \emptyset, \emptyset, \ldots, 0, \emptyset, \emptyset\rangle$;
- $X = \{x_1, \ldots, x_m\}$;
- $\mathcal{F} = \{F_1, \ldots, F_m\}$ where F_j contains all locations with $\ell_j = 0$ or $\ell_j = 2$;
- Δ is obtained by synchronising (in the same way as before) $\langle \ell, S, T\rangle \xrightarrow{\sigma, g, \lambda} \langle \ell', S', T'\rangle$ of individual sub-components:
  - $p_\psi \notin \sigma$; $\ell' = 0$, $\ell = 0$; $S' = \emptyset$, $S = \emptyset$, $T' = \emptyset$, $T = \emptyset$; $g = \top$; $\lambda = \emptyset$.
  - $p_\psi \notin \sigma$; $\ell' = 1$, $\ell \in \{1, 2\}$; $S \xrightarrow[\vee]{\sigma} S'$, $T' = \emptyset$, $T = \emptyset$; $g = \top$; $\lambda = \emptyset$.
  - $p_\psi \notin \sigma$; $\ell' = 3$, $\ell = 3$; $S \xrightarrow[\vee]{\sigma} S'$, $T \xrightarrow[\vee]{\sigma} T'$; $g = \top$; $\lambda = \emptyset$.
  - $p_\psi \notin \sigma$; $\ell' = 0$, $\ell \in \{1, 2\}$; $S' = \emptyset$, $S = \{s^{\mathcal{A}}\}$, $s_F^{\mathcal{A}} \models \delta(s^{\mathcal{A}}, \sigma)$ for some $s_F^{\mathcal{A}} \in F^{\mathcal{A}}$, $T' = \emptyset$, $T = \emptyset$; $g = x \geq c$; $\lambda = \emptyset$.
  - $p_\psi \notin \sigma$; $\ell' = 2$, $\ell = 3$; $T \xrightarrow[\vee]{\sigma} S'$, $S = \{s^{\mathcal{A}}\}$, $s_F^{\mathcal{A}} \models \delta(s^{\mathcal{A}}, \sigma)$ for some $s_F^{\mathcal{A}} \in F^{\mathcal{A}}$, $T' = \emptyset$; $g = x \geq c$; $\lambda = \{x\}$.
  - $p_\psi \in \sigma$; $\ell' = 1$, $\ell = 0$; $\{s_0^{\mathcal{A}}\} \xrightarrow[\vee]{\sigma} S'$, $S = \emptyset$, $T' = \emptyset$, $T = \emptyset$; $g = \top$; $\lambda = \{x\}$.
  - $p_\psi \in \sigma$; $\ell' = 1$, $\ell \in \{1, 2\}$; S' is the union of some S'' such that $S \xrightarrow[\vee]{\sigma} S''$ and S''' with $\{s_0^{\mathcal{A}}\} \xrightarrow[\vee]{\sigma} S'''$, $T' = \emptyset$, $T = \emptyset$; $g = \top$; $\lambda = \{x\}$.
  - $p_\psi \in \sigma$; $\ell' = 3$, $\ell = 1$; $S \xrightarrow[\vee]{\sigma} S'$, $\{s_0^{\mathcal{A}}\} \xrightarrow[\vee]{\sigma} T'$, $T = \emptyset$; $g = \top$; $\lambda = \emptyset$.
  - $p_\psi \in \sigma$; $\ell' = 3$, $\ell = 3$; S' is the union of some T'' such that $T \xrightarrow[\vee]{\sigma} T''$ and T''' with $\{s_0^{\mathcal{A}}\} \xrightarrow[\vee]{\sigma} T'''$; $g = \top$; $\lambda = \emptyset$.
  - $p_\psi \in \sigma$; $\ell' = 2$, $\ell \in \{1, 2\}$; $\{s_0^{\mathcal{A}}\} \xrightarrow[\vee]{\sigma} S'$, $S = \{s^{\mathcal{A}}\}$, $s_F^{\mathcal{A}} \models \delta(s^{\mathcal{A}}, \sigma)$ for some $s_F^{\mathcal{A}} \in F^{\mathcal{A}}$, $T' = \emptyset$, $T = \emptyset$; $g = x \geq c$; $\lambda = \{x\}$.
  - $p_\psi \in \sigma$; $\ell' = 2$, $\ell = 3$; S' is the union of some S'' such that $T \xrightarrow[\vee]{\sigma} S''$ and S''' with $\{s_0^{\mathcal{A}}\} \xrightarrow[\vee]{\sigma} S'''$, $S = \{s^{\mathcal{A}}\}$, $s_F^{\mathcal{A}} \models \delta(s^{\mathcal{A}}, \sigma)$ for some $s_F^{\mathcal{A}} \in F^{\mathcal{A}}$, $T' = \emptyset$; $g = x \geq c$; $\lambda = \{x\}$.

Proposition 4.3. $[\![C^\psi]\!] = [\![\mathbf{G}\left(p_\psi \Rightarrow \mathcal{A}_{\geq c}(\overline{\varphi_1}, \ldots, \overline{\varphi_n})\right)]\!]$.

Proof sketch. Similar to the proof of Proposition 4.2, but since $[c, \infty)$ is *upward closed*, we replace all the subtrees rooted at nodes $(s^{\mathcal{A}}, v, i)$ with $(s^{\mathcal{A}}, v^{min}, i)$, where $v^{min} = \min\{v \mid (s^{\mathcal{A}}, v) \in L_i\}$; in C^ψ, we still put the obligations that correspond to nodes $(s^{\mathcal{A}}, v, i)$ into the sub-component that holds the (oldest) obligation that corresponds to $(s^{\mathcal{A}}, v^{max}, i)$. There is, however, a potential issue: since we reset x_j whenever the trigger p_ψ is pulled and C_j^ψ is chosen, it might be the case that x_j never reaches c, i.e. the satisfaction of the obligations in S_j are delayed indefinitely. Following [17], we solve this by locations with $\ell_j = 3$ such that, when entered, we stop resetting x_j and put the new obligations into T_j instead; when the obligations in S_j are fulfilled, we move the obligations in T_j to S_j and reset x_j. The other direction obviously holds as each sub-component C_j^ψ resets x_j when $p_\psi \in \sigma$ and $\ell_j \in \{1, 2\}$, unless

it goes from $\ell_j = 1$ to $\ell_j = 3$. In other words, C^ψ_j adds the new obligation to S_j while strengthening the existing ones in S_j. □

$\psi = \tilde{\mathcal{A}}_{\leq c}$. Let $C^\psi = \langle \Sigma, S, s_0, X, \Delta, \mathcal{F} \rangle$ be defined as follows:

- Each location $s \in S$ is of the form $\langle S_1, \ldots, S_{m+1} \rangle$ where $S_j \subseteq S^{\mathcal{A}}$ for all $j \in \{1, \ldots, m+1\}$; intuitively, S_j can be seen as a location of the sub-component C^ψ_j;
- $s_0 = \langle \emptyset, \ldots, \emptyset \rangle$;
- $X = \{x_1, \ldots, x_{m+1}\}$;
- $\mathcal{F} = \emptyset$, i.e. any run is accepting;
- Δ is obtained by synchronising (in the same way as before) $S \xrightarrow{\sigma, g, \lambda} S'$ of individual sub-components:
  - $p_\psi \notin \sigma; S' = \emptyset, S = \emptyset; g = \top; \lambda = \emptyset$.
  - $p_\psi \notin \sigma; S \xrightarrow[\wedge]{\sigma} S', S' \cap F^{\mathcal{A}} = \emptyset; g = x \leq c; \lambda = \emptyset$.
  - $p_\psi \notin \sigma; S' = \emptyset; g = x > c; \lambda = \emptyset$.
  - $p_\psi \in \sigma; \{s_0^{\mathcal{A}}\} \xrightarrow[\wedge]{\sigma} S', S' \cap F^{\mathcal{A}} = \emptyset, S = \emptyset; g = \top; \lambda = \{x\}$.
  - $p_\psi \in \sigma; S'$ is the union of some S'' such that $S \xrightarrow[\wedge]{\sigma} S''$ and S''' with $\{s_0^{\mathcal{A}}\} \xrightarrow[\wedge]{\sigma} S''', S' \cap F^{\mathcal{A}} = \emptyset; g = x \leq c; \lambda = \{x\}$.
  - $p_\psi \in \sigma; \{s_0^{\mathcal{A}}\} \xrightarrow[\wedge]{\sigma} S', S' \cap F^{\mathcal{A}} = \emptyset; g = x > c; \lambda = \{x\}$.

Proposition 4.4. $[\![C^\psi]\!] = [\![\mathrm{G}\,(p_\psi \Rightarrow \tilde{\mathcal{A}}_{\leq c}(\overline{\varphi_1}, \ldots, \overline{\varphi_n}))]\!]$.

Proof sketch. Let $\psi' = \mathrm{G}\,(p_\psi \Rightarrow \tilde{\mathcal{A}}_{\leq c}(\overline{\varphi_1}, \ldots, \overline{\varphi_n}))$ and $\mathcal{A}_{\psi'}$ be the equivalent OCATA obtained via Proposition 4.1. Consider each level $L_i = \{(s, v) \mid (s, v, i) \in V\}$ of an accepting run $G = \langle V, \rightarrow \rangle$ of $\mathcal{A}_{\psi'}$ on $\rho = (\sigma_i, \tau_i)_{i \geq 1}$ in the increasing order of i. In C^ψ, whenever the trigger p_ψ is pulled, we attempt to put the corresponding obligation into an unused sub-component (with $S = \emptyset$) or a sub-component that can be cleared (with $x > c$); if this is not possible, since for every $s^{\mathcal{A}} \in S^{\mathcal{A}}$ all the subtrees rooted at nodes $(s^{\mathcal{A}}, v, i)$ can be replaced with the subtree rooted at $(s^{\mathcal{A}}, v^{min}, i)$ where $v^{min} = \min\{v \mid (s^{\mathcal{A}}, v) \in L_i\}$, at least one sub-component C^ψ_j becomes redundant, i.e. all of its obligations are implied by the other sub-components C^ψ_k, $k \neq j$. A consequence is that the obligations in the sub-component C^ψ_k with the minimal non-negative value of $x_j - x_k$ can be merged with the obligations in C^ψ_j, freeing up a sub-component for the current incoming obligation. This can be repeated to construct an accepting run of C^ψ on ρ. The other direction holds as each C^ψ_j adds the new obligation to S_j while strengthening the existing obligations in S_j. □

$\psi = \tilde{\mathcal{A}}_{\geq c}$. Let $C^\psi = \langle \Sigma, S, s_0, X, \Delta, \mathcal{F} \rangle$ be defined as follows (for simplicity, assume that $c > 0$):

- Each location $s \in S$ is of the form $\langle S_1, T_1 \ldots, S_{m+1}, T_{m+1} \rangle$ where $S_j, T_j \subseteq S^{\mathcal{A}}$ for all $j \in \{1, \ldots, m+1\}$; intuitively, $\langle S_j, T_j \rangle$ can be seen as a location of the sub-component C^ψ_j;
- $s_0 = \langle \emptyset, \emptyset, \ldots, \emptyset, \emptyset \rangle$;
- $X = \{x_1, \ldots, x_{m+1}\}$;
- $\mathcal{F} = \emptyset$, i.e. any run is accepting;
- Δ is obtained by synchronising (in the same way as before) $\langle \ell, S, T \rangle \xrightarrow{\sigma, g, \lambda} \langle \ell', S', T' \rangle$ of individual sub-components:
  - $p_\psi \notin \sigma; S' = \emptyset, S = \emptyset, T' = \emptyset, T = \emptyset; g = \top; \lambda = \emptyset$.
  - $p_\psi \notin \sigma; S' = \emptyset, S = \emptyset, T \xrightarrow[\wedge]{\sigma} T', T' \cap F^{\mathcal{A}} = \emptyset; g = \top; \lambda = \emptyset$.
  - $p_\psi \notin \sigma; S \xrightarrow[\wedge]{\sigma} S', T' = \emptyset, T = \emptyset; g = x < c; \lambda = \emptyset$.
  - $p_\psi \notin \sigma; S \xrightarrow[\wedge]{\sigma} S', T \xrightarrow[\wedge]{\sigma} T', T' \cap F^{\mathcal{A}} = \emptyset; g = x < c; \lambda = \emptyset$.
  - $p_\psi \notin \sigma; S' = \emptyset, T'$ is the union of some T'' such that $S \xrightarrow[\wedge]{\sigma} T''$ and T''' with $T \xrightarrow[\wedge]{\sigma} T''', T' \cap F^{\mathcal{A}} = \emptyset; g = x \geq c; \lambda = \emptyset$.
  - $p_\psi \in \sigma; \{s_0^{\mathcal{A}}\} \xrightarrow[\wedge]{\sigma} S', S = \emptyset, T' = \emptyset, T = \emptyset; g = \top; \lambda = \{x\}$.
  - $p_\psi \in \sigma; \{s_0^{\mathcal{A}}\} \xrightarrow[\wedge]{\sigma} S', S = \emptyset, T \xrightarrow[\wedge]{\sigma} T', T' \cap F^{\mathcal{A}} = \emptyset; g = \top; \lambda = \{x\}$.
  - $p_\psi \in \sigma; S'$ is the union of some S'' such that $S \xrightarrow[\wedge]{\sigma} S''$ and S''' with $\{s_0^{\mathcal{A}}\} \xrightarrow[\wedge]{\sigma} S''', T' = \emptyset, T = \emptyset; g = x < c; \lambda = \emptyset$.
  - $p_\psi \in \sigma; S'$ is the union of some S'' such that $S \xrightarrow[\wedge]{\sigma} S''$ and S''' with $\{s_0^{\mathcal{A}}\} \xrightarrow[\wedge]{\sigma} S''', T \xrightarrow[\wedge]{\sigma} T', T' \cap F^{\mathcal{A}} = \emptyset; g = x < c; \lambda = \emptyset$.
  - $p_\psi \in \sigma; \{s_0^{\mathcal{A}}\} \xrightarrow[\wedge]{\sigma} S', T'$ is the union of some T'' such that $S \xrightarrow[\wedge]{\sigma} T''$ and T''' with $T \xrightarrow[\wedge]{\sigma} T''', T' \cap F^{\mathcal{A}} = \emptyset; g = x \geq c; \lambda = \{x\}$.

Proposition 4.5. $[\![C^\psi]\!] = [\![\mathrm{G}\,(p_\psi \Rightarrow \tilde{\mathcal{A}}_{\geq c}(\overline{\varphi_1}, \ldots, \overline{\varphi_n}))]\!]$.

Proof sketch. As in the proof of Proposition 4.4, whenever p_ψ is pulled in C^ψ, we attempt to put the corresponding obligation into an unused sub-component (with $S = \emptyset$) or a sub-component that can be cleared (if $x > c$, we move the obligations in S to T and let them remain there). If this is not possible, since for every $s^{\mathcal{A}} \in S^{\mathcal{A}}$ all the subtrees rooted at nodes $(s^{\mathcal{A}}, v, i)$ can be replaced with the subtree rooted at $(s^{\mathcal{A}}, v^{max}, i)$ where $v^{max} = \max\{v \mid (s^{\mathcal{A}}, v) \in L_i\}$, some C^ψ_j becomes redundant, and the obligations in the sub-component C^ψ_k with the minimal non-negative value of $x_k - x_j$ can be merged with the obligations in C^ψ_j, freeing up a sub-component for the current incoming obligation. This can be repeated to construct an accepting run of C^ψ on ρ. The other direction holds as each C^ψ_j adds an obligation that is at least as strong to S_j without weakening the existing obligations in S_j. □

Finally, thanks to the fact that each location of C^ψ can be represented using space polynomial in the size of $\mathcal{A}$, and the product $C^{init} \times \prod_{\psi \in \Phi} C^\psi$ need not to be constructed explicitly, we can state the main result of this section.

Theorem 4.6. *The satisfiability and model-checking problems for* $\mathsf{EMITL}_{0,\infty}$ *over timed words are* PSPACE*-complete.*

Corollary 4.7. *The satisfiability and model-checking problems for* EECL *over timed words are* PSPACE*-complete.*

5 CONCLUSION

It is shown that $\mathsf{EMITL}_{0,\infty}$ and EECL are already as expressive as EMITL over timed words, a somewhat unexpected yet very pleasant result. We also provided a compositional construction from $\mathsf{EMITL}_{0,\infty}$ to diagonal-free TAs based on one-clock alternating timed automata (OCATAs); this allows satisfiability and model checking based on existing algorithmic back ends for TAs. The natural next step would be to implement the construction and evaluate its performance on real-world use cases. Another possible future direction is to investigate whether similar techniques can be used to handle full EMITL or larger fragments of OCATAs (like [21]).

ACKNOWLEDGMENTS

The author would like to thank Thomas Brihaye, Chih-Hong Cheng, Thomas Ferrère, Gilles Geeraerts, Timothy M. Jones, Arthur Milchior, and Benjamin Monmege for their help and fruitful discussions. The author would also like to thank the anonymous reviewers for their comments. This work is supported by the Engineering and Physical Sciences Research Council (EPSRC) through grant EP/P020011/1 and (partially) by F.R.S.-FNRS PDR grant SyVeRLo.

REFERENCES

[1] Houssam Abbas, Alena Rodionova, Ezio Bartocci, Scott A. Smolka, and Radu Grosu. 2017. Quantitative Regular Expressions for Arrhythmia Detection Algorithms. In *CMSB (LNCS)*, Vol. 10545. Springer, 23–39.

[2] Rajeev Alur and David L. Dill. 1994. A Theory of Timed Automata. *Theoretical Computer Science* 126, 2 (1994), 183–235.

[3] Rajeev Alur, Tomás Feder, and Thomas A. Henzinger. 1996. The Benefits of Relaxing Punctuality. *J. ACM* 43, 1 (1996), 116–146.

[4] Rajeev Alur and Thomas A. Henzinger. 1993. Real-Time Logics: Complexity and Expressiveness. *Information and Computation* 104, 1 (1993), 35–77.

[5] Rajeev Alur and Thomas A. Henzinger. 1994. A Really Temporal Logic. *J. ACM* 41, 1 (1994), 164–169.

[6] Roy Armoni, Limor Fix, Alon Flaisher, Rob Gerth, Boris Ginsburg, Tomer Kanza, Avner Landver, Sela Mador-Haim, Eli Singerman, Andreas Tiemeyer, Moshe Y. Vardi, and Yael Zbar. 2002. The ForSpec Temporal Logic: A New Temporal Property Specification Language. In *TACAS (LNCS)*, Vol. 2280. Springer, 296–311.

[7] Eugene Asarin, Paul Caspi, and Oded Maler. 1997. A Kleene Theorem for Timed Automata. In *LICS*. IEEE, 160–171.

[8] Eugene Asarin, Paul Caspi, and Oded Maler. 2002. Timed regular expressions. *J. ACM* 49, 2 (2002), 172–206.

[9] Behnam Banieqbal and Howard Barringer. 1987. Temporal Logic with Fixed Points. In *TLS (LNCS)*, Vol. 398. Springer, 62–74.

[10] David A. Basin, Srdan Krstic, and Dmitriy Traytel. 2017. Almost Event-Rate Independent Monitoring of Metric Dynamic Logic. In *RV (LNCS)*, Vol. 10548. Springer, 85–102.

[11] Gerd Behrmann, Alexandre David, Kim Guldstrand Larsen, John Håkansson, Paul Pettersson, Wang Yi, and Martijn Hendriks. 2006. UPPAAL 4.0. In *QEST*. IEEE, 125–126.

[12] Patricia Bouyer. 2003. Untameable Timed Automata!. In *STACS (LNCS)*, Vol. 2607. Springer, 620–631.

[13] Patricia Bouyer and Fabrice Chevalier. 2005. On Conciseness of Extensions of Timed Automata. *Journal of Automata, Languages and Combinatorics* 10, 4 (2005), 393–405.

[14] Patricia Bouyer, Fabrice Chevalier, and Nicolas Markey. 2010. On the expressiveness of TPTL and MTL. *Information and Computation* 208, 2 (2010), 97–116.

[15] Patricia Bouyer, François Laroussinie, and Pierre-Alain Reynier. 2005. Diagonal Constraints in Timed Automata: Forward Analysis of Timed Systems. In *FORMATS (LNCS)*, Vol. 3829. Springer, 112–126.

[16] Thomas Brihaye, Morgane Estiévenart, and Gilles Geeraerts. 2014. On MITL and Alternating Timed Automata of Infinite Words. In *FORMATS (LNCS)*, Vol. 8711. Springer.

[17] Thomas Brihaye, Gilles Geeraerts, Hsi-Ming Ho, and Benjamin Monmege. 2017. MightyL: A Compositional Translation from MITL to Timed Automata. In *CAV (LNCS)*, Vol. 10426. Springer, 421–440.

[18] Thomas Brihaye, Gilles Geeraerts, Hsi-Ming Ho, and Benjamin Monmege. 2017. Timed-Automata-Based Verification of MITL over Signals. In *TIME (LIPIcs)*, Vol. 90. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 7:1–7:19.

[19] Costas Courcoubetis, Moshe Y. Vardi, Pierre Wolper, and Mihalis Yannakakis. 1992. Memory-Efficient Algorithms for the Verification of Temporal Properties. *Formal Methods in System Design* 1, 2/3 (1992), 275–288.

[20] Cindy Eisner and Dana Fisman. 2006. *A Practical Introduction to PSL*. Springer. 1–240 pages.

[21] Thomas Ferrère. 2018. The Compound Interest in Relaxing Punctuality. In *FM (LNCS)*. Springer. To appear.

[22] Dov Gabbay, Amir Pnueli, Sharanon Shelah, and J. Stavi. 1980. On the Temporal Analysis of Fairness. In *Proceedings of POPL 1980*. ACM Press, 163–173.

[23] P. Gastin, S. Mukherjee, and B Srivathsan. 2018. Reachability in timed automata with diagonal constraints. *CoRR* abs/1806.11007 (2018). arXiv:1806.11007 http://arxiv.org/abs/1806.11007

[24] Paul Gastin and Denis Oddoux. 2001. Fast LTL to Büchi Automata Translation. In *CAV (LNCS)*, Vol. 2102. Springer, 53–65.

[25] VIRES Simulationstechnologie GmbH. 2016. OpenSCENARIO - Bringing content to the road. (2016). http://www.openscenario.org/docs/OSCUserMeeting20160629pub.pdf Accessed: 2018-09-01.

[26] Thomas A. Henzinger, Jean-François Raskin, and Pierre-Yves Schobbens. 1998. The Regular Real-Time Languages. In *ICALP (LNCS)*, Vol. 1443. Springer, 580–591.

[27] Yoram Hirshfeld and Alexander Rabinovich. 1999. A framework for decidable metrical logics. In *ICALP (LNCS)*, Vol. 1644. Springer, 422–432.

[28] Yoram Hirshfeld and Alexander Rabinovich. 2007. Expressiveness of Metric modalities for continuous time. *Logical Methods in Computer Science* 3, 1 (2007), 1–11.

[29] Johan A. Kamp. 1968. *Tense logic and the theory of linear order*. Ph.D. Dissertation. University of California, Los Angeles.

[30] Gijs Kant, Alfons Laarman, Jeroen Meijer, Jaco van de Pol, Stefan Blom, and Tom van Dijk. 2015. LTSmin: High-Performance Language-Independent Model Checking. In *TACAS (LNCS)*, Vol. 9035. Springer, 692–707.

[31] Yonit Kesten, Amir Pnueli, and Li on Raviv. 1998. Algorithmic Verification of Linear Temporal Logic Specifications. In *ICALP (LNCS)*, Vol. 1443. Springer, 1–16.

[32] Ron Koymans. 1990. Specifying Real-time Properties with Metric Temporal Logic. *Real-Time Systems* 2, 4 (1990), 255–299.

[33] Shankara Narayanan Krishna, Khushraj Madnani, and Paritosh K. Pandya. 2016. Metric Temporal Logic with Counting. In *FoSSaCS (LNCS)*, Vol. 9634. Springer, 335–352.

[34] Shankara Narayanan Krishna, Khushraj Madnani, and Paritosh K. Pandya. 2017. Making Metric Temporal Logic Rational. In *MFCS (LIPIcs)*, Vol. 83. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 77:1–77:14.

[35] Shankara Narayanan Krishna, Khushraj Madnani, and Paritosh K. Pandya. 2018. Büchi-Kamp Theorems for 1-clock ATA. *CoRR* abs/1802.02514 (2018). arXiv:1802.02514 http://arxiv.org/abs/1802.02514

[36] Orna Kupferman and Moshe Y. Vardi. 1997. Weak Alternating Automata Are Not That Weak. In *ISTCS*. IEEE, 147–158.

[37] David E. Muller, Ahmed Saoudi, and Paul E. Schupp. 1986. Alternating Automata, the Weak Monadic Theory of the Tree, and its Complexity. In *ICALP (LNCS)*, Vol. 226. Springer, 275–283.

[38] Dejan Ničković. 2008. *Checking Timed and Hybrid Properties: Theory and Applications*. Ph.D. Dissertation. VERIMAG.

[39] Dejan Ničković, Olivier Lebeltel, Oded Maler, Thomas Ferrère, and Dogan Ulus. 2018. AMT 2.0: Qualitative and Quantitative Trace Analysis with Extended Signal Temporal Logic. In *TACAS (LNCS)*, Vol. 10806. Springer, 303–319.

[40] Joël Ouaknine and James Worrell. 2006. On Metric Temporal Logic and Faulty Turing Machines. In *FoSSaCS (LNCS)*, Vol. 3921. Springer, 217–230.

[41] Joël Ouaknine and James Worrell. 2007. On the Decidability and Complexity of Metric Temporal Logic over Finite Words. *Logical Methods in Computer Science* 3, 1 (2007).

[42] Amir Pnueli. 1977. The temporal logic of programs. In *FOCS*. IEEE, 46–57.

[43] Jean-François Raskin. 1999. *Logics, automata and classical theories for deciding real time*. Ph.D. Dissertation. FUNDP (Belgium).

[44] Gareth Scott Rohde. 1997. *Alternating automata and the temporal logic of ordinals*. Ph.D. Dissertation. University of Illinois, Urbana Campaign.

[45] A. P. Sistla and E. M. Clarke. 1985. The Complexity of Propositional Linear Temporal Logics. *J. ACM* 32, 3 (1985), 733–749.

[46] A. Prasad Sistla, Moshe Y. Vardi, and Pierre Wolper. 1985. The Complementation Problem for Büchi Automata with Applications to Temporal Logic (Extended Abstract). In *ICALP (LNCS)*, Vol. 194. Springer, 465–474.

[47] Moshe Y. Vardi. 1987. Unified Verification Theory. In *TLS (LNCS)*, Vol. 398. Springer, 202–212.

[48] Thomas Wilke. 1994. Specifying timed state sequences in powerful decidable logics and timed automata. In *FTRTFT (LNCS)*, Vol. 863. Springer, 694–715.

[49] Pierre Wolper. 1983. Temporal Logic Can be More Expressive. *Information and Control* 56, 1/2 (1983), 72–99.
[50] Pierre Wolper and Moshe Y. Vardi. 1994. Reasoning about Infinite Computations. *Information and Computation* 115, 1 (1994), 1–37.

A PROOF OF PROPOSITION ??

Proof. We first show that $[\![\mathcal{A}_\varphi]\!] \subseteq [\![\varphi]\!]$. Suppose that $\mathcal{A}_\varphi = \langle \Sigma, S, s_0, \delta, F\rangle$ has an accepting run $G = \langle V, \to\rangle$ on $\rho = (\sigma_1, \tau_1)(\sigma_2, \tau_2)\cdots$ and let $L_i = \{(s, v) \mid (s, v, i) \in V\}$. We claim that:

(1) For each $s^{\mathcal{A}} \in S^{\mathcal{A}}$ where $\mathcal{A}$ occurs in a subformula $\mathcal{A}_I(\varphi_1, \ldots, \varphi_n)$ and $i \geq 1$, $L_i \models_v s^{\mathcal{A}}$ implies that there is a *finite* rooted DAG $G' = \langle V', \to'\rangle$ with $V' \subseteq S^{\mathcal{A}} \times \mathbb{N}_{\geq 0}$, $(s^{\mathcal{A}}, i)$ as the root, and for each vertex (s, k) either (a) $k > i$, $v+(\tau_k - \tau_i) \in I$, and $s \in F^{\mathcal{A}}$, or (b) there is some $a_{k+1} \in \Sigma^{\mathcal{A}}$ such that $L_{k+1} \models_{v+(\tau_{k+1}-\tau_i)} \delta(\varphi_{a_{k+1}}, \sigma_{k+1})$ and there is a model M of the formula $\delta^{\mathcal{A}}(s, a_{k+1})$ such that $(s, k) \to' (s', k+1)$ for every state s' in M.
(2) For each $s^{\mathcal{A}} \in S^{\mathcal{A}}$ where $\mathcal{A}$ occurs in a subformula $\tilde{\mathcal{A}}_I(\varphi_1, \ldots, \varphi_n)$ and $i \geq 1$, $L_i \models_v s^{\mathcal{A}}$ implies that there is a rooted DAG $G' = \langle V', \to'\rangle$ with $V' \subseteq S^{\mathcal{A}} \times \mathbb{N}_{\geq 0}$, $(s^{\mathcal{A}}, i)$ as the root, and each vertex (s, k) satisfies (a) if $k > i$ then $v + (\tau_k - \tau_i) \in I$ implies $s \notin F^{\mathcal{A}}$, and (b) for every $a_{k+1} \in \Sigma^{\mathcal{A}}$, either $L_{k+1} \models_{v+(\tau_{k+1}-\tau_i)} \delta(\varphi_{a_{k+1}}, \sigma_{k+1})$ or there is a model M of the formula $\overline{\delta^{\mathcal{A}}}(s, a_{k+1})$ such that $(s, k) \to' (s', k+1)$ for every state s' in M.

We prove (1); (2) can be proved similarly. First note that every branch labelled with locations of $\mathcal{A}$ must terminate. If $(s^{\mathcal{A}}, v, i)$ is a leaf with respect to $S^{\mathcal{A}}$ in G, we must have $N \models_{v+(\tau_{i+1}-\tau_i)} \delta(\varphi_a, \sigma_{i+1}) \wedge \delta^{\mathcal{A}}[s_F^{\mathcal{A}} \leftarrow s_F^{\mathcal{A}} \vee x \in I](s^{\mathcal{A}}, a)$ for some $a \in \Sigma^{\mathcal{A}}$ and $N \subseteq (S \setminus S^{\mathcal{A}}) \times \mathbb{R}_{\geq 0}$ such that $N \subseteq L_{i+1}$. It follows that either (i) $N \models_{v+(\tau_{i+1}-\tau_i)} \delta(\varphi_a, \sigma_{i+1}) \wedge \delta^{\mathcal{A}}(s^{\mathcal{A}}, a)$ or (ii) $v+(\tau_{i+1}-\tau_i) \in I$ and $N \cup M \models_{v+(\tau_{i+1}-\tau_i)} \delta(\varphi_a, \sigma_{i+1}) \wedge \delta^{\mathcal{A}}(s^{\mathcal{A}}, a)$ for some nonempty $M \subseteq F^{\mathcal{A}} \times \{v + (\tau_{i+1} - \tau_i)\}$. In case (i), (b) trivially holds. In case (ii) note that $N \models_{v+(\tau_{i+1}-\tau_i)} \delta(\varphi_a, \sigma_{i+1})$, and let $\{(s', i+1) \mid (s', v') \in M\}$ be the successors of $(s^{\mathcal{A}}, i)$ in G': each of them satisfies (a). If $(s^{\mathcal{A}}, v, i)$ is not a leaf with respect to $S^{\mathcal{A}}$ in G, we have $N \cup M \models_{v+(\tau_{i+1}-\tau_i)} \delta(\varphi_a, \sigma_{i+1}) \wedge \delta^{\mathcal{A}}[s_F^{\mathcal{A}} \leftarrow s_F^{\mathcal{A}} \vee x \in I](s^{\mathcal{A}}, a)$ for some $a \in \Sigma^{\mathcal{A}}$, $N \subseteq (S \setminus S^{\mathcal{A}}) \times \mathbb{R}_{\geq 0}$ such that $N \subseteq L_{i+1}$, and nonempty $M \subseteq S^{\mathcal{A}} \times \{v+(\tau_{i+1}-\tau_i)\}$ such that $M \subseteq L_{i+1}$. It follows that either (i) $N \cup M \models_{v+(\tau_{i+1}-\tau_i)} \delta(\varphi_a, \sigma_{i+1}) \wedge \delta^{\mathcal{A}}(s^{\mathcal{A}}, a)$ or (ii) $v+(\tau_{i+1}-\tau_i) \in I$ and $N \cup M' \models_{v+(\tau_{i+1}-\tau_i)} \delta(\varphi_a, \sigma_{i+1}) \wedge \delta^{\mathcal{A}}(s^{\mathcal{A}}, a)$ for some $M' \supset M$ with $M' \setminus M \subseteq F^{\mathcal{A}} \times \{v + (\tau_{i+1} - \tau_i)\}$. In case (i), let $\{(s', i+1) \mid (s', v') \in M\}$ be the successors of $(s^{\mathcal{A}}, i)$ in G'; applying the IH on $L_{i+1} \models_{v+(\tau_{i+1}-\tau_i)} s'$ for all such s' yields the corresponding sub-DAGs, which we combine to obtain G'. Case (ii) is similar.

We now claim that for each subformula ψ of φ and $i \geq 1$, $L_i \models_0 \delta(\psi, \sigma_i)$ implies $\rho, i \models \psi$. The cases of atomic propositions, negation, and Boolean operators are trivial. For $\psi = \mathcal{A}_I(\varphi_1, \ldots, \varphi_n)$, as $L_i \models_0 \delta(s_0^{\mathcal{A}}, \sigma_i)$ we have, for some $a \in \Sigma^{\mathcal{A}}$, (i) $L_i \models_0 \delta(\varphi_a, \sigma_i)$ and (ii) $L_i \models_0 \delta^{\mathcal{A}}[s_F^{\mathcal{A}} \leftarrow s_F^{\mathcal{A}} \vee x \in I](s_0^{\mathcal{A}}, a)$. From (i) and the IH we obtain $\rho, i \models \varphi_a$. From (ii) we deduce that there is $M \subseteq (S^{\mathcal{A}} \setminus F^{\mathcal{A}}) \times \{0\}$ such that $M \subseteq L_i$, and $M' \supseteq M$ such that $M' \setminus M \subseteq F^{\mathcal{A}} \times \{0\}$ and $M' \models_0 \delta^{\mathcal{A}}(s_0^{\mathcal{A}}, a)$. Applying (1) on every $s^{\mathcal{A}} \in M$ and the IH (note that by the definition of δ, we have $L_{k+1} \models_0 \delta(\varphi_{a_{k+1}}, \sigma_{k+1})$ if $L_{k+1} \models_{v+(\tau_{k+1}-\tau_i)} \delta(\varphi_{a_{k+1}}, \sigma_{k+1})$) and combining sub-DAGs gives $\rho, i \models \psi$. The case of $\psi = \tilde{\mathcal{A}}_I(\varphi_1, \ldots, \varphi_n)$ is analogous.

Finally, as $s^{init} \in L_0$, we have $L_1 \models_{\tau_1} x.\delta(\varphi, \sigma_1)$, which is equivalent to $L_1 \models_0 \delta(\varphi, \sigma_1)$. It follows that $\rho, 1 \models \varphi$; this finishes the proof of $[\![\mathcal{A}_\varphi]\!] \subseteq [\![\varphi]\!]$.

For the other direction, observe that $\mathcal{A}_{\neg\varphi} = (\mathcal{A}_\varphi)^c$ (up to renaming of locations) where $(\mathcal{A}_\varphi)^c$ is obtained from $\mathcal{A}_\varphi$ by dualising the transition function and swapping the sets of final and non-final locations. The desired result ($[\![\varphi]\!] \subseteq [\![\mathcal{A}_\varphi]\!]$) immediately follows from the fact that $\mathcal{A}_\varphi$ is a *tree-like* OCATA [16]. □