The Compound Interest in Relaxing Punctuality*

Thomas Ferrère

IST Austria

Abstract. Imprecision in timing can sometimes be beneficial: Metric interval temporal logic (MITL), disabling the expression of punctuality constraints, was shown to translate to timed automata, yielding an elementary decision procedure. We show how this principle extends to other forms of dense-time specification using regular expressions. By providing a clean, automaton-based formal framework for non-punctual languages, we are able to recover and extend several results in timed systems. Metric interval regular expressions (MIRE) are introduced, providing regular expressions with non-singular duration constraints. We obtain that MIRE are expressively complete relative to a class of one-clock timed automata, which can be determinized using additional clocks. Metric interval dynamic logic (MIDL) is then defined using MIRE as temporal modalities. We show that MIDL generalizes known extensions of MITL, while translating to timed automata at comparable cost.

1 Introduction

Regular expressions (RE) [20] are a basic notion in computer science. They provide a simple algebraic way to describe finite-state behaviors. Since their introduction in verification and testing, alongside linear temporal logic (LTL) [32], regular expressions have also proven to be a very practical formalism to specify discrete systems behavior [14,36]. Yet not all applications enjoy the synchronous, discrete-time style of modeling captured by finite automata. Modern computerized systems are more asynchronous in nature, calling for a different level of abstraction in which time may no longer be discrete.

Timed automata (TA) [2] are widely regarded as a natural extension of finite-state theory to dense-time. This model of computation uses real-valued variables known as *clocks* to control delays between events. The strength of timed automata, beyond the simplicity of their definition, comes from their theoretical properties: the emptiness problem is solvable in polynomial space, timed regular languages are closed under positive Boolean operations, and their *untiming* yields back regular languages. However the standard, nondeterministic model (NTA) is not closed under complement, while the deterministic model (DTA) is not closed under concatenation or Kleene star.

Negation is a desirable operation in any specification language. Metric temporal logic (MTL) [21] is a well-studied, established dense-time specification

* This research was supported by the Austrian Science Fund (FWF) under grants S11402-N23 (RiSE/SHiNE) and Z211-N23 (Wittgenstein Award).

language. Through negation, the set of languages described in MTL is closed under complement. However satisfiability of MTL is non-elementary under the hypotheses of [30], and undecidable in general [3,31]. Timed regular expressions (TRE) [6] constitute an interesting alternative to MTL, both powerful and intuitive. The emptiness of TRE is also decidable in polynomial space, since TRE translate to timed automata in polynomial time [6]. But TRE do not feature a negation operator, which would render them undecidable.

Virtually all negative results in timed systems, such as the undecidability of language inclusion for timed automata, rely on the ability to enforce real delays with infinite precision—some extreme form of *punctuality*. When no semantic restriction is placed on the variability or duration of behaviors, a single unit of time can hold an arbitrary amount of information, which can then be repeatedly transfered from one time unit to the next, encoding Turing computations. A standard way to regain decidability is to bound the variability of behaviors [16,28]. Another, less conventional way is to bound their duration [29].

The syntactic restriction of [3] simply bounds the precision timing constraints—in effect *relaxing punctuality*. Decidability of the resulting metric interval temporal logic (MITL) [3] follows, by translation to timed automata. Subsequently, extensions of MITL with finite automata [37,17] and threshold counting [18] have then been proposed, enjoying special connections with monadic logic [37,19,17]. In this context, our contribution consists in (a) the definition of RE-based variants of MITL for specifying timed behaviors; (b) a simple automaton-based framework in which several results regarding these variants can be derived (Figure 1).

In particular, we show how to adapt the subset construction of [10] to determinize arbitrary control structures, by introducing the notion of *metric interval automaton* (MIA, Section 3). These automata, reminiscent of [5], have a single clock, checked against non-singular timing intervals, and reset after every check. A simple state-elimination argument demonstrates that this model is equivalent to the proposed *metric interval regular expressions* (MIRE, Section 3), which therefore translate to deterministic timed automata (Section 4). By treating metric interval automata as modalities, we redefine *extended MITL* (EMITL, [37]). Building on our initial results, we propose *metric interval dynamic logic* (MIDL, Section 5), equivalent in expressive power, and provide a translation to non-deterministic timed automata (Section 6). This translation is compositional, in the style of [26], and does not go through intermediate formalisms such as monadic logic [37] or *event clock* automata [17].

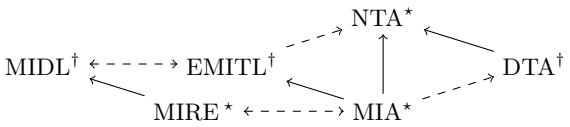

Fig. 1. Translations (⇢) and inclusions (⟶) between formalisms. Closure under Boolean operations (†) and under regular operations (⋆) are indicated in exponent.

2 Preliminaries

In this section, we introduce basic definitions and relevant results. We take the time domain $\mathbb{T} = \mathbb{R}_{\geq 0}$ to be the non-negative reals. Given a set of times $R \subseteq \mathbb{T}$, we write $\mathrm{ch}(R) = \{t \in \mathbb{T} \mid \exists r, r' \in R, r \leq t \leq r'\}$ its convex hull and $t \oplus R = \{t + r \in \mathbb{T} \mid r \in R\}$ its Minkowski sum with some $t \in \mathbb{T}$. We define *timed words* as sequences alternating delays in $\mathbb{T}$ and events in some alphabet Σ. Given a timed word $w = t_1 a_1 \dots t_n a_n$ we write $w_{i..j}$ its infix $t_{i+1} a_{i+1} \dots t_j a_j$ between positions $0 \leq i \leq j \leq n$. We denote by $|w| = n$ the size of w and by $\|w\| = \sum_{i=1}^{n} t_i$ the *duration* of w. The empty word ϵ verifies $|\epsilon| = \|\epsilon\| = 0$.

Automata. Following [2], automata are equipped with a set X of clock variables. A *clock constraint* is a Boolean combination of inequalities of the form $x \bowtie c$, or $x - y \bowtie c$, where $c \in \mathbb{N}$ is a constant, $\bowtie \in \{\leq, <, >, \geq\}$ is a comparison sign, and $x, y \in X$ are clocks. The set of clock constraints over X is denoted $\Phi(X)$. A *valuation* v associates any clock variable $x \in X$ with a delay $v(x) \in \mathbb{T}$. We write $v \models \phi$ when the constraint ϕ is satisfied under clock valuation v.

A *timed automaton* is a tuple $\mathcal{A} = (\Sigma, X, L, S, F, \Delta)$ where L is a set of locations, $S \subseteq L$ is a set of initial locations, $F \subseteq L$ is a set of accepting locations, and $\Delta \subseteq L \times \Sigma \times \Phi(X) \times 2^X \times L$ is a set of edges. A *state* of $\mathcal{A}$ is a pair (ℓ, v) where ℓ is a location in L and v is a valuation over X. For delays $t \in \mathbb{T}$ and events $a \in \Sigma$, transitions $\overset{t}{\rightsquigarrow}$ and $\overset{a}{\to}$ in $\mathcal{A}$ are defined as the following relations:

- $(\ell, v) \overset{t}{\rightsquigarrow} (\ell', v')$ if $\ell = \ell'$ and $v' = v + t$;
- $(\ell, v) \overset{a}{\to} (\ell', v')$ if $v \models \phi$ and $v' = v[Z \leftarrow 0]$ for some $(\ell, a, \phi, Z, \ell') \in \Delta$.

Here $v + t$ stands for the valuation such that $(v + t)(x) = v(x) + t$ for all $x \in X$, and $v[Z \leftarrow 0]$ stands for the valuation such that $v[Z \leftarrow 0](x) = 0$ if $x \in Z$, $v(x)$ otherwise. A *run* of automaton $\mathcal{A}$ over the word $w = t_1 a_1 \dots t_n a_n$ is a sequence $(\ell_0, v_0) \overset{t_1}{\rightsquigarrow} (\ell_0, v'_0) \xrightarrow{a_1} \dots \overset{t_n}{\rightsquigarrow} (\ell_{n-1}, v'_{n-1}) \xrightarrow{a_n} (\ell_n, v_n)$ of transitions labeled by delays and events in w such that $\ell_0 \in S$, and $v_0(x) = 0$ for all $x \in X$. The language $\mathcal{L}(\mathcal{A})$ is the set of words over which there exists a run of $\mathcal{A}$ ending in an accepting location. We say that $\mathcal{A}$ is deterministic when $S = \{\ell_0\}$ for some ℓ_0, and $\phi_1 \wedge \phi_2$ is unsatisfiable for all $(\ell, a, \phi_1, Z_1, \ell_1) \neq (\ell, a, \phi_2, Z_2, \ell_2) \in \Delta$.

Expressions. We define timed regular expressions (TRE) following [6], but without intersection or projection. They are given by the grammar:

$$\varphi ::= \epsilon \mid a \mid \varphi \cup \varphi \mid \varphi \cdot \varphi \mid \varphi^* \mid \varphi_I$$

where $a \in \Sigma$, and $I \subseteq \mathbb{T}$ is an integer-bounded interval. As customary iterating an expression φ is denoted in exponent, with by convention $\varphi^+ \equiv \varphi^* \cdot \varphi$, and $\varphi^k \equiv \epsilon$ if $k = 0$, $\varphi^k \equiv \varphi^{k-1} \cdot \varphi$ otherwise. Any TRE φ can be associated with a language $\mathcal{L}(\varphi)$ defined inductively as follows:

$$\mathcal{L}(\epsilon) = \{\epsilon\} \qquad \mathcal{L}(\varphi_1 \cdot \varphi_2) = \{w_1 w_2 \mid w_1 \in \mathcal{L}(\varphi_1),\, w_2 \in \mathcal{L}(\varphi_2)\}$$

$$\mathcal{L}(a) = \{ta \mid t \in \mathbb{T}\} \qquad \mathcal{L}(\varphi^*) = \bigcup_{k=0}^{\infty} \mathcal{L}(\varphi^k)$$

$$\mathcal{L}(\varphi_1 \cup \varphi_2) = \mathcal{L}(\varphi_1) \cup \mathcal{L}(\varphi_2) \qquad \mathcal{L}(\varphi_I) = \{w \mid w \in \mathcal{L}(\varphi),\, \|w\| \in I\}.$$

The *size* of a TRE φ is the number of atomic expressions it contains. Its *depth* $\mathrm{d}(\varphi)$ is the level of nesting of timing constraints in φ, defined by $\mathrm{d}(a) = \mathrm{d}(\epsilon) = 0$, $\mathrm{d}(\varphi \cdot \psi) = \mathrm{d}(\varphi \cup \psi) = \max\{\mathrm{d}(\varphi), \mathrm{d}(\psi)\}$, $\mathrm{d}(\varphi^*) = \mathrm{d}(\varphi)$, and $\mathrm{d}(\varphi_I) = \mathrm{d}(\varphi) + 1$.

Theorem 1 (TRE ⇒ NTA, [6]). *For any TRE of size m and depth n, one can construct an equivalent timed automaton with n clocks and $m + 1$ locations.*

Logic. Metric temporal logic (MTL) [21] extends LTL [32] by providing the *until* operator with a timing interval. MTL formulas are given by the grammar:

$$\psi ::= a \mid \psi \vee \psi \mid \neg\psi \mid \psi \,\mathcal{U}_I\, \psi$$

where $a \in \Sigma$ and I is an integer-bounded interval. Operators *eventually* and *always* are defined by letting $\Diamond_I \varphi \equiv \top \,\mathcal{U}_I\, \varphi$ and $\Box_I \varphi \equiv \neg \Diamond_I \neg\varphi$. The timing interval $[0, \infty)$ is usually omitted as subscript. Metric interval temporal logic (MITL) [3] is the fragment of MTL where intervals I are *non-singular* ($\inf I < \sup I$).

The semantics $\models$ of MTL and MITL is defined over *pointed words*, pairs (w, i) of timed word w and position $0 < i \leq |w| + 1$, as follows:

$$\begin{array}{lll}
(w,i) \models a & \text{iff} & w_{i-1..i} = ta \text{ for some } t \in \mathbb{T} \\
(w,i) \models \neg\psi & \text{iff} & (w,i) \not\models \psi \\
(w,i) \models \psi_1 \vee \psi_2 & \text{iff} & (w,i) \models \psi_1 \text{ or } (w,i) \models \psi_2 \\
(w,i) \models \psi_1 \,\mathcal{U}_I\, \psi_2 & \text{iff} & (w,j) \models \psi_2 \text{ for some } j > i \text{ such that } \|w_{i..j}\| \in I \\
 & & \text{and } (w,k) \models \psi_1 \text{ for all } i < k < j.
\end{array}$$

The language $\mathcal{L}(\psi)$ of formula ψ is defined by $\mathcal{L}(\psi) = \{w \mid (w, 1) \models \psi\}$. The *size* of an MITL formula ψ is the number of temporal operators it contains. Its *resolution* $\mathrm{r}(\psi)$ is the maximal relative interval width in ψ, defined by $\mathrm{r}(a) = 0$, $\mathrm{r}(\psi_1 \vee \psi_2) = \max\{\mathrm{r}(\psi_1), \mathrm{r}(\psi_2)\}$, $\mathrm{r}(\neg\psi) = \mathrm{r}(\psi)$, and $\mathrm{r}(\psi_1 \,\mathcal{U}_I\, \psi_2) = \max\{\mathrm{r}(\psi_1), \mathrm{r}(\psi_2), \mathrm{r}(\mathcal{U}_I)\}$, where $\mathrm{r}(\mathcal{U}_I) = \left\lfloor \frac{\sup I}{\sup I - \inf I} \right\rfloor + 2$ if $\sup I < \infty$, 1 otherwise.

Theorem 2 (MITL ⇒ NTA, [3]). *For any MITL formula of size m and resolution n, one can construct an equivalent timed automaton with $2mn$ clocks and 2^{8mn+1} locations.*

3 Metric Interval Regular Expressions

We now introduce *metric interval regular expressions* (MIRE) as TRE of depth 1 and devoid of singular timing intervals. Formally, they are given by the grammar:

$$\begin{aligned}
\varphi &::= \gamma_I \mid \varphi \cdot \varphi \mid \varphi \cup \varphi \mid \varphi^* \\
\gamma &::= \epsilon \mid a \mid \gamma \cdot \gamma \mid \gamma \cup \gamma \mid \gamma^*
\end{aligned}$$

where $a \in \Sigma$ and I is a non-singular, integer-bounded interval. Timing interval $[0, \infty)$ is usually omitted, that is, we write γ in place of $\gamma_{[0,\infty)}$ in MIRE. The *resolution* of a MIRE is defined similarly as for MITL.

Example 1. Consider the expression $(a \cup b \cdot (a^* \cdot b)_{[2,3]})^*$. It describes sequences of events a and b in which every odd occurrence of b is followed by another (even) occurrence of b within 2 to 3 time units.

Automaton Model. We define *metric interval automata* (MIA) as timed automata with a single clock x in which every edge $(\ell, a, \phi, Z, \ell')$ is such that either $Z = \emptyset$ and $\phi = \top$, or $Z = \{x\}$ and $\phi \equiv x \in I$ for some non-singular interval I. Here $x \in I$ is the abbreviated notation for constraint $x \geq c$ when $I = [c, \infty)$, $x \geq c \wedge x \leq d$ when $I = [c, d]$, and similar when I is a (semi-)open interval.

Proposition 1 (MIRE $\Leftrightarrow$ MIA). *Every MIRE language is recognizable by MIA, and every MIA language is expressible as MIRE.*

Direction $\Rightarrow$ is a refinement of Theorem 1, and will be proved in Section 4. We treat direction $\Leftarrow$ in two steps. Let $\mathcal{A}$ be a MIA. Assume without loss of generality that locations of $\mathcal{A}$ are partitioned into two sets L_0 and L_1, such that edges to L_0 reset the clock while edges to L_1 don't, and initial and final locations of $\mathcal{A}$ lie in L_0. First, we remove all locations in L_1, using the state removal technique in finite automata [35]. This yields an equivalent MIA $\mathcal{A}'$ whose edges are labeled by regular expressions instead of events. Second, we remove clock resets and constraints from $\mathcal{A}'$ by replacing every edge $(\ell, \gamma, x \in I, \{x\}, \ell')$ with $(\ell, (\gamma)_I, \top, \emptyset, \ell')$. We obtain a finite automaton with MIRE labels. We perform again standard state removal to eliminate intermediate locations in L_0. The resulting automaton has only one edge, labeled by a MIRE equivalent to $\mathcal{A}$.

Comparison with MITL. Following Proposition 1, all MIRE properties can be checked using one clock. In contrast, some MITL properties require more than one clock, even when using nondeterminism. For instance the formula $\Box(a \to \Diamond_{[1,2]}\, b)$ requires two clocks [25]. In the other direction MIRE feature *untimed* modulo-counting languages, such as $(a^2)^*$, not expressible in MITL. More interestingly, MIRE also feature additional *timed* properties.

Example 2. Consider the expression $\chi \equiv a \cdot ((a^+)_{[1,2]})^+$ over the alphabet $\{a\}$. It describes words featuring an extracted timed word, skipping an arbitrary number of intermediate a events and accumulating delays, such that consecutive events in the extracted timed word are separated by 1 to 2 time units.

We show similarly as in [18] that the language of χ in Example 2 cannot be expressed in MITL. For this, define a family of words (w_n) as follows: $w_0 = \epsilon$, and $w_n = \frac{3}{4}\, a\, w_{n-1}$ for all $n > 0$. Observe that $w_n \in \mathcal{L}(\chi)$ iff $n > 1$ and n is odd, as illustrated in Figure 2. In contrast for every MITL formula ψ there exist a bound k such that $w_n \in \mathcal{L}(\psi)$ iff $w_{n+1} \in \mathcal{L}(\psi)$, for all $n \geq k$. This bound is straightforward to obtain by structural induction. Thus, $\mathcal{L}(\chi) \neq \mathcal{L}(\psi)$.

4 From MIRE to Deterministic Timed Automata

In this section, we show that MIRE translate to deterministic timed automata. The first step of the procedure translates a MIRE φ into an equivalent MIA $\mathcal{A}_\varphi$ in a standard way. The second step performs some kind of subset construction to turn $\mathcal{A}_\varphi$ into a deterministic automaton $\mathcal{A}'_\varphi$. Because timed automata have a bounded number of clocks, over a given timed word automaton $\mathcal{A}'_\varphi$ cannot store

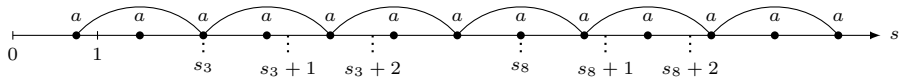

Fig. 2. Timed word w_{13}, with events a occurring at absolute times $s = s_1, s_2, \ldots, s_{13}$. Expression $\chi \equiv a \cdot ((a^+)_{[1,2]})^+$ entails only one possible decomposition of w_{13} as shown. Events at times s_i for i even do not appear in this decomposition but are locally indistinguishable from those at s_j for j odd.

the set of possible states of $\mathcal{A}_\varphi$ explicitly. To this effect we adapt the notion of approximation of [10] to group in intervals possible clock values in $\mathcal{A}_\varphi$ that have a similar future. We show the soundess of this approximation, and demonstrate how it can be implemented in a deterministic automaton.

Translation to MIA. Automaton $\mathcal{A}_\varphi = (\Sigma, X_\varphi, L_\varphi, S_\varphi, \{\ell_\varphi\}, \Delta_\varphi)$ equivalent to the MIRE φ is obtained by structural induction. We assume that automata given by induction hypothesis have disjoint sets of locations but they may share the same clock x.

- Atomic expressions: $\mathcal{A}_\epsilon$ has its final location ℓ_ϵ marked as initial, no edge and no clock; $\mathcal{A}_a$ further has one edge labeled a from ℓ_ϵ to its final location ℓ_a.
- Disjunction: $\mathcal{A}_{\varphi\cup\psi}$ is obtained by replacing ℓ_φ and ℓ_ψ with $\ell_{\varphi\cup\psi}$ in the component-wise union of $\mathcal{A}_\varphi$ and $\mathcal{A}_\psi$.
- Concatenation: $\mathcal{A}_{\varphi\cdot\psi}$ is defined by letting $X_{\varphi\cdot\psi} = X_\varphi \cup X_\psi$, $L_{\varphi\cdot\psi} = L_\varphi \cup L_\psi \setminus \{\ell_\varphi\}$, $S_{\varphi\cdot\psi} = S_\varphi$ if $\ell_\varphi \notin S_\varphi$, $S_{\varphi\cdot\psi} = S_\varphi \cup S_\psi \setminus \{\ell_\varphi\}$ otherwise, and $\ell_{\varphi\cdot\psi} = \ell_\psi$. The set $\Delta_{\varphi\cdot\psi}$ is obtained from $\Delta_\varphi \cup \Delta_\psi$ by replacing every edge $(\ell, a, \phi, Z, \ell_\varphi)$ with edges $(\ell, a, \phi, X_\psi, \ell')$ for all $\ell' \in S_\psi$.
- Kleene star: without loss of generality, assume that φ^+ is primitive and φ^* is derived as $\epsilon \cup \varphi^+$. Define $\mathcal{A}_{\varphi^+}$ by letting $X_{\varphi^+} = X_\varphi$, $L_{\varphi^+} = L_\varphi$, $S_{\varphi^+} = S_\varphi$, $\ell_{\varphi^+} = \ell_\varphi$, and $\Delta_{\varphi^+} = \Delta_\varphi \cup \{(\ell, a, \phi, X_\varphi, \ell') \mid (\ell, a, \phi, Z, \ell_\varphi) \in \Delta_\varphi, \ell' \in S_\varphi\}$.
- Duration constraint: $\mathcal{A}_{\gamma_I}$ is defined by $X_{\gamma_I} = \{x\}$, $L_{\gamma_I} = L_\gamma$, $S_{\gamma_I} = S_\gamma$ if $0 \in I$, $S_{\gamma_I} = S_\gamma \setminus \{\ell_\gamma\}$ otherwise, and $\ell_{\gamma_I} = \ell_\gamma$. The set Δ_{γ_I} is obtained from Δ_γ by replacing every edge $(\ell, a, \top, \emptyset, \ell_\gamma)$ with $(\ell, a, x \in I, \{x\}, \ell_\gamma)$.

Example 2 (Continued). The expression $\chi \equiv a \cdot ((a^+)_{[1,2]})^+$ translates to the automaton $\mathcal{A}_\chi$ depicted in Figure 3.

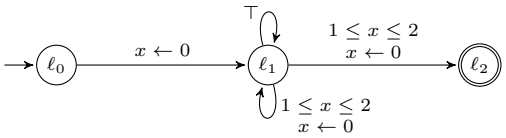

Fig. 3. Automaton $\mathcal{A}_\chi$ translating χ (event labels are omitted).

Parallel Runs. Fix $\mathcal{A}$ a metric interval automaton with clock x. We now treat valuations of x as real values $t \in \mathbb{T}$, and introduce the following definitions. An *interval state* (ℓ, J) pairs a location ℓ with an interval J, representing the set of states $\{(\ell, t) \mid t \in J\}$. A *configuration* is a set of interval states. Transition functions $\overset{t}{\rightsquigarrow}, \xrightarrow{a}$ between configurations C, D of $\mathcal{A}$ are such that $C \overset{t}{\rightsquigarrow} D$ iff $D = \{(\ell, t \oplus J) \mid (\ell, J) \in C\}$, and $C \xrightarrow{a} D$ iff $D = \{(\ell', J') \mid \exists (\ell, J) \in C, (\ell, J) \xrightarrow{a} (\ell', J')\}$. Here, by $(\ell, J) \xrightarrow{a} (\ell', J')$ we mean that $\mathcal{A}$ has an edge of the form $(\ell, a, \phi, Z, \ell')$ such that $t \models \phi$ for at least one $t \in J$, and $J = J'$ if $Z = \emptyset$, $J = \{0\}$ otherwise. The *parallel run* of automaton $\mathcal{A}$ over some word $w = t_1 a_1 \dots t_n a_n$ is a sequence of transitions $C_0 \overset{t_1}{\rightsquigarrow} C'_0 \xrightarrow{a_1} \dots \overset{t_n}{\rightsquigarrow} C'_{n-1} \xrightarrow{a_n} C_n$ labeled by w, where the *initial* configuration C_0 is the set of interval states $(\ell, [0, 0])$ for ℓ initial. All intervals appearing in a parallel run are singular.

Lemma 1. *There exists a run of $\mathcal{A}$ over w finishing in a given location ℓ iff the final configuration of the parallel run of $\mathcal{A}$ over w features ℓ.*

$C_0 \overset{0.5}{\rightsquigarrow} \xrightarrow{a} \{(\ell_1, [0, 0])\}$

$\overset{1.3}{\rightsquigarrow} \xrightarrow{a} \{(\ell_1, [0, 0]),$
$(\ell_1, [1.3, 1.3]), (\ell_2, [0, 0])\}$

$\overset{0.3}{\rightsquigarrow} \xrightarrow{a} \{(\ell_1, [0, 0]), (\ell_1, [0.3, 0.3]),$
$(\ell_1, [1.6, 1.6]), (\ell_2, [0, 0])\}$

$\overset{0.2}{\rightsquigarrow} \xrightarrow{a} \{(\ell_1, [0, 0]), (\ell_1, [0.2, 0.2]),$
$(\ell_1, [0.5, 0.5]), (\ell_1, [1.8, 1.8]), (\ell_2, [0, 0])\}$

$\overset{0.9}{\rightsquigarrow} \xrightarrow{a} \{(\ell_1, [0, 0]), (\ell_1, [0.9, 0.9]), (\ell_1, [1.1, 1.1]),$
$(\ell_1, [1.4, 1.4]), (\ell_1, [2.7, 2.7]), (\ell_2, [0, 0])\}$

$C_0 \overset{0.5}{\rightsquigarrow} \xrightarrow{a} \xrightarrow{\prec} \{(\ell_1, [0, 0])\}$

$\overset{1.3}{\rightsquigarrow} \xrightarrow{a} \xrightarrow{\prec} \{(\ell_1, [0, 0]),$
$(\ell_1, [1.3, 1.3]), (\ell_2, [0, 0])\}$

$\overset{0.3}{\rightsquigarrow} \xrightarrow{a} \xrightarrow{\prec} \{(\ell_1, [0, 0.3]),$
$(\ell_1, [1.6, 1.6]), (\ell_2, [0, 0])\}$

$\overset{0.2}{\rightsquigarrow} \xrightarrow{a} \xrightarrow{\prec} \{(\ell_1, [0, 0.5]),$
$(\ell_1, [1.8, 1.8]), (\ell_2, [0, 0])\}$

$\overset{0.9}{\rightsquigarrow} \xrightarrow{a} \xrightarrow{\prec} \{(\ell_1, [0, 1.4]),$
$(\ell_1, [2.7, 2.7]), (\ell_2, [0, 0])\}$

Fig. 4. The parallel and $\prec$-parallel runs of $\mathcal{A}_\chi$ over w.

Example 2 (Continued). Consider timed word $w = 0.5\, a\, 1.3\, a\, 0.3\, a\, 0.2\, a\, 0.9\, a$ and automaton $\mathcal{A}_\chi$. The parallel run of $\mathcal{A}_\chi$ over w is shown in the left part of Figure 4. Since ℓ_2 appears in the final configuration, we have $w \in \mathcal{L}(\mathcal{A}_\chi)$.

Approximation. We now define an approximation relation (to be correct, a *simulation* relation) between configurations closely matching the one in [10]. Let c, d stand respectively for the largest b and smallest $b - a$ across clock constraints in $\mathcal{A}$ of the form $x \triangleright a \wedge x \triangleleft b$ for some $\triangleright \in \{>, \geq\}$ and $\triangleleft \in \{<, \leq\}$. In the absence of such constraints, take $c = 0$ and $d = \infty$. Approximation relation $\prec$ over configurations will be used to merge intervals either less than d apart, or extend beyond c. It is defined by letting $C \prec D$ when $C \setminus \{(\ell, I), (\ell, J)\} = D \setminus \{(\ell, \mathrm{ch}(I \cup J))\}$ for some $(\ell, I) \neq (\ell, J) \in C$, $(\ell, \mathrm{ch}(I \cup J)) \in D$ such that $\inf J - \sup I < d$ and $\inf I - \sup J < d$, or $\sup(I \cup J) > c$. When all clock constraints are strict (resp. non-strict) we can use $\geq c$ (resp. $\leq d$) instead.

Approximate Parallel Runs. Let us now write $C \xrightarrow{\prec}\!\!\!\!\rightarrow D$ when D is maximal relative to $\prec$ such that $C \prec^* D$, where $*$ denotes reflexive-transitive closure. A *$\prec$-parallel run* of automaton $\mathcal{A}$ over some word $w = t_1 a_1 \dots t_n a_n$ is a sequence of transitions $C_0 \overset{t_1}{\rightsquigarrow} C_0' \xrightarrow{a_1} C_0'' \xrightarrow{\prec}\!\!\!\!\rightarrow \dots \overset{t_n}{\rightsquigarrow} C_{n-1}' \xrightarrow{a_n} C_{n-1}'' \xrightarrow{\prec}\!\!\!\!\rightarrow C_n$ labeled by w interleaved with approximations, from the initial configuration C_0. Relation $\prec$ constitutes a faithful abstraction in the sense of the following lemma.

Lemma 2. *For any word w, the set of locations that appear in final configurations of the parallel, and $\prec$-parallel runs of $\mathcal{A}$ over w, are the same.*

The approximation behaves deterministically: for any configuration C of $\mathcal{A}$ there is a unique D such that $C \xrightarrow{\prec}\!\!\!\!\rightarrow D$. It also ensures the size of configurations also stays bounded. Let $m = |L|$ be the number of locations of $\mathcal{A}$, and let n be the *resolution* of $\mathcal{A}$, defined by $n = \left\lfloor \frac{c}{d} \right\rfloor + 2$ if $d < \infty$, 1 otherwise. For any configurations $C \xrightarrow{\prec}\!\!\!\!\rightarrow D$ of $\mathcal{A}$, we have $|D| \leq mn$.

Example 2 (Continued). The $\prec$-parallel run of $\mathcal{A}_\chi$ over w, shown in the right part of Figure 4, groups clock values stemming from events number 2 to 5 in w. We check that ℓ_2 appears in the final configuration, and $w \in \mathcal{L}(\mathcal{A}_\chi)$.

Subset Construction. We translate a given MIA $\mathcal{A} = (\Sigma, \{x\}, L, S, F, \Delta)$ to the deterministic timed automaton $\mathcal{A}' = (\Sigma, X', L', S', F', \Delta')$ as follows.

- Clocks: $X' = Y \cup Y'$ with $Y = \{y_1, y_2, \dots, y_{mn}\}$, $Y' = \{y_1', y_2', \dots, y_{mn}'\}$.
- Locations: $L' = 2^{L \times Y \times Y'}$.
- Initial locations: $S' = \{Q_0\}$, where $Q_0 = S \times \{y_1\} \times \{y_1'\}$.
- Accepting locations: $F' = \{Q \in L' \mid Q \cap (F \times Y \times Y') \neq \emptyset\}$.
- Edges: Δ' is built as follows. For every source $P \in L'$, letter a, feasible set of edges $E \subseteq \Delta$, and potential target $Q \in L'$, we construct:
  - constraint $\theta(P, E)$ ensuring that E is exactly the set of edges of $\mathcal{A}$ that can be taken from P;
  - configuration $R(P, E)$ reached when taking such edges;
  - constraint $\lambda_\prec(R, Q)$ ensuring that Q approximates R.

  Edges from P to Q are guarded by the conjunction of θ and $\lambda_\prec$, and reset either no clock, one clock in Y, or a pair of clocks in $Y \times Y'$.

Given a valuation v, clock pair $yy' \in Y \times Y'$ represents the interval $[v(y), v(y')]$, location $Q \in L'$ represents the configuration $v(Q) = \{(\ell, [v(y), v(y')]) \mid \ell yy' \in Q\}$.

Edges. We now present in detail the construction of Δ'. For $yy' \in Y \times Y'$ and $\phi \in \Phi(\{x\})$ let $\phi[yy']$ stand for the constraint ϕ in which y (resp. y') replaces x in upper (resp. lower) bound comparisons. For any valuation v with $v(y) < v(y')$, we have $v \models \phi[yy']$ iff there exists $t \in [v(y), v(y')]$ such that $t \models \phi$. Now let $P \in L'$ and $a \in \Sigma$. Denote by $\Delta(P, a) \subseteq \Delta$ the set of edges labeled a and whose source location appears in P. Given a subset $E \subseteq \Delta(P, a)$, we define the constraint $\theta(P, E)$ ensuring that edges fired from P upon event a are precisely those in E:

$$\theta(P, E) \equiv \bigwedge\nolimits_{\ell yy' \in P, (\ell, a, \phi, Z, \ell') \in E} \phi[yy'] \wedge \bigwedge\nolimits_{\ell yy' \in P, (\ell, a, \phi, Z, \ell') \in \Delta(P,a) \setminus E} \neg\phi[yy'].$$

Clock resets are temporarily handled using fresh variables y_0 and y'_0, extending sets of clocks to $Y_0 = Y \cup \{y_0\}$, $Y'_0 = Y' \cup \{y'_0\}$ and set of locations to $L'_0 = 2^{L \times Y_0 \times Y'_0}$. The target configuration $R(P, E) \in L'_0$ when firing edges in E from P is defined by letting

$$R(P,E) = \{\ell' yy' \mid \ell yy' \in P,\ (\ell, a, \top, \emptyset, \ell') \in E\} \cup \{\ell' y_0 y'_0 \mid (\ell, a, \phi, \{x\}, \ell') \in E\}.$$

When $\theta(P, E)$ holds, automaton $\mathcal{A}'$ transits to a configuration that approximates $R(P, E)$. Given configurations $Q, R \in L'_0$, we now define $\lambda_\prec(R, Q)$ ensuring that Q approximates R. We would like that $v \models \lambda_\prec(R, Q)$ iff $v(R) \xrightarrow{\prec} v(Q)$, for all valuations v. But if some clocks share the same value, for a given R there may be more than one Q such that $v(R) \xrightarrow{\prec} v(Q)$. Priority is given to clocks with lowest index. Given indices $i, i', j, j' \in \{0, \ldots, mn\}$, $k \in \{i, j\}$ and $k' \in \{i', j'\}$, define

$$\begin{aligned}\mu_{ii'jj'kk'} \equiv ((y_i - y'_{j'} < c \wedge y_j - y'_{i'} < c) \vee (y_i > d \wedge y_j > d)) \wedge \\ (y_i > y_k \vee (y_i = y_k \wedge i \leq k)) \wedge ((y_j > y_k \vee y_j = y_k) \wedge j \leq k) \wedge \\ (y'_{i'} < y'_{k'} \vee (y'_{i'} = y'_{k'} \wedge i' \leq k')) \wedge (y'_{j'} < y'_{k'} \vee (y'_{j'} = y'_{k'} \wedge j' \leq k')).\end{aligned}$$

For any $\ell \in L$, constraint $\mu_{ii'jj'kk'}$ ensures that $\ell y_i y'_{i'}$ and $\ell y_j y'_{j'}$ should be merged to $\ell y_k y'_{k'}$. The constraint $\lambda_\prec(R, Q)$ is defined as the conjunction of two parts: (1) the disjunction over well-formed chains of merges $i_1 i'_1 j_1 j'_1 k_1 k'_1, \ldots, i_h i'_h j_h j'_h k_h k'_h$ from R to Q of conjunctions of μ over the chains; (2) the conjunction of $\neg\mu$ over all possible merges in Q. This guarantees that one such chain is (1) correct and (2) maximal in length. We can now replace temporary variables y_0, y'_0 with available clocks in $Y \cup Y'$. Let us define the set of clocks Z_Q as follows:

- If both y_0 and y'_0 occur in Q, let $Z_Q = \{y_i, y'_{i'}\}$ for $i, i' \geq 1$ the least indices such that $y_i, y'_{i'}$ do not occur in Q;
- If y_0 occurs in Q but not y'_0, let $Z_Q = \{y_i\}$ for $i \geq 1$ the least index such that y_i does not occur in Q;
- Otherwise, let $Z_Q = \emptyset$.

We write $\overline{Q} \in L$ to denote the configuration Q in which y_0, y'_0 are replaced by clocks in Z_Q. The set of edges of $\mathcal{A}'$ is obtained by letting

$$\Delta' = \{(P, a, \theta(P,E) \wedge \lambda_\prec(R(P,E), Q), Z_Q, \overline{Q}) \mid P \in L', a \in \Sigma, Q \in L'_0, E \subseteq \Delta(P, a)\}.$$

Theorem 3 (MIRE ⇒ DTA). *For any MIRE of size m and resolution n, one can construct an equivalent deterministic timed automaton with $2mn$ clocks and $2^{m^3n^2+1}$ locations.*

Example 2 (Continued). Applying the above procedure to $\mathcal{A}_\chi$, we obtain automaton $\mathcal{A}'_\chi$ of Figure 5. We use the following simplifications. In $\mathcal{A}_\chi$, any state in location ℓ_1 with clock value above 2 cannot reach ℓ_2. We remove interval states $\ell yy'$ from target configurations of $\mathcal{A}'_\chi$ for any $y \in Y$ such that $y > 2$. Transitions preserve the ordering of non-reset clocks, and we use this to simplify clock constraints. Locations not (co-)reachable are also removed.

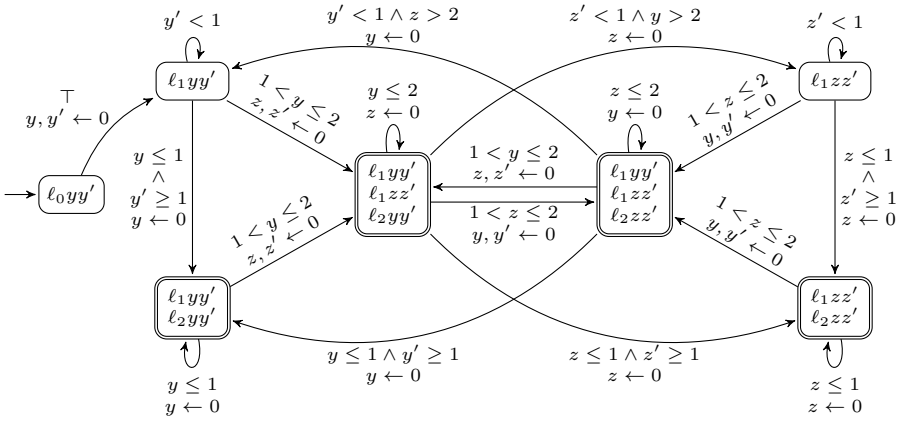

Fig. 5. Automaton $\mathcal{A}'_\chi$ determinizing $\mathcal{A}_\chi$ (event labels are omitted).

5 Metric Interval Dynamic Logic

We now introduce *metric interval dynamic logic* (MIDL) as the dynamic logic of MIRE. It provides linear dynamic logic (LDL) [13,15] with timing constraints.

Syntax. MIDL formulas ψ and expressions φ are given by the grammar

$$\begin{aligned}\psi &::= a \mid \neg\psi \mid \psi \vee \psi \mid \langle\varphi\rangle\, \psi \\ \varphi &::= \gamma_I \mid \varphi \cdot \varphi \mid \varphi \cup \varphi \mid \varphi^* \\ \gamma &::= \epsilon \mid \psi? \mid \gamma \cdot \gamma \mid \gamma \cup \gamma \mid \gamma^*\end{aligned}$$

where $a \in \Sigma$ and I is a non-singular integer-bounded interval. The *size* of an MIDL formula is the total size of expressions φ in its modalities $\langle\varphi\rangle$. The *resolution* of MIDL formulas is defined inductively as for MITL.

The form $\langle\varphi\rangle\, \psi$ is known as *suffix conjunction* and is satisfied when ψ holds at some future time instant such that φ matches the events from now to that time instant. When φ is of the form γ_I for simplicity we write $\langle\gamma\rangle_I\, \psi$ in place of $\langle\gamma_I\rangle\, \psi$. Observe that $\langle\varphi_1 \cdot \varphi_2\rangle\, \psi \Leftrightarrow \langle\varphi_1\rangle\, \langle\varphi_2\rangle\, \psi$ and $\langle\varphi_1 \cup \varphi_2\rangle\, \psi \Leftrightarrow \langle\varphi_1\rangle\, \psi \vee \langle\varphi_2\rangle\, \psi$, hence when no star is applied to a timed subexpression, formulas can be rewritten using modalities of the form $\langle\gamma\rangle_I$ only. The form $\psi?$ is known as a *test* and matches any time instant where ψ holds. We also write a in place of $a?$ for any $a \in \Sigma$. Using this convention, MIRE are a fragment of MIDL.

Semantics. The semantics $\models$ of MIDL formulas is defined over pointed words, with the same inductive definitions as MITL in the case of events $a \in \Sigma$ and Boolean connectives $\neg$, $\vee$. The case of suffix implication $\langle\varphi\rangle$ is as follows:

$$(w, i) \models \langle\varphi\rangle\, \psi \qquad \text{iff} \qquad (w, i, j) \Vdash \varphi \text{ and } (w, j) \models \psi \text{ for some } j \geq i.$$

The semantics $\models$ of MIDL expressions is defined over *bi-pointed words*, triples (w, i, j) of timed word w and positions $0 \leq i \leq j \leq |w|$, as follows.

$$
\begin{array}{lll}
(w,i,j) \models \epsilon & \text{iff} & j = i \\
(w,i,j) \models \psi? & \text{iff} & j = i+1 \text{ and } (w,j) \models \psi \\
(w,i,j) \models \varphi_1 \cdot \varphi_2 & \text{iff} & (w,i,k) \models \varphi_1 \text{ and } (w,k,j) \models \varphi_2 \text{ for some } k \\
(w,i,j) \models \varphi_1 \cup \varphi_2 & \text{iff} & (w,i,j) \models \varphi_1 \text{ or } (w,i,j) \models \varphi_2 \\
(w,i,j) \models \varphi^* & \text{iff} & (w,i,j) \models \varphi^k \text{ for some } k \\
(w,i,j) \models \varphi_I & \text{iff} & (w,i,j) \models \varphi \text{ and } \|w_{i..j}\| \in I.
\end{array}
$$

This semantics definition is compatible with that of MIRE and TRE in general. The language $\mathcal{L}(\psi)$ of formula ψ is defined by $\mathcal{L}(\psi) = \{w \mid (w,1) \models \psi\}$.

Temporal Logic. The *until* operator can be defined in MIDL as the abbreviation $\psi_1 \mathcal{U}_I \psi_2 \equiv \langle \psi_1 ?^* \cdot \top ? \rangle_I \psi_2$. We also use operators *always* and *eventually* as previously. In general MIDL is more expressive than MITL.

Example 3. Consider the formula $\xi \equiv \Box a \rightarrow \langle \top ?^* \cdot b \cdot \top ?^+ \rangle_{(0,1)} c)$ over the alphabet $\{a, b, c, d\}$. It describes words in which every occurrence of a triggers in the future within less than one time unit an occurrence of b followed by one of c.

A conjecture of [4], proved in [9], states that formulas similar to the one above cannot be expressed in MITL. Replacing b, c by arbitrary formulas, we obtain an instance of so-called *Pnueli modality* [18]. The simpler *threshold counting* modalities $\langle (\top ?^* \cdot \varphi ?)^{k-1} \cdot \top ?^+ \rangle_I \varphi$, requiring that φ holds k times within I time units, already cannot be expressed in MITL for $k > 1$, see [18].

Automata Modalities. Let us define *extended MITL* (EMITL) by adding to MITL the syntactic clause $\psi ::= \mathcal{A}(\psi_1, \ldots, \psi_m)$, where $\mathcal{A}$ is a metric interval automaton over the alphabet $2^{\{\psi_1, \ldots, \psi_m\}}$. The semantics of this clause is such that $(w, i) \models \mathcal{A}(\psi_1, \ldots, \psi_m)$ iff the word $t_{i+1} \Psi_{i+1} \ldots t_n \Psi_n$ is accepted by $\mathcal{A}$, where each t_j is the j-th delay in w and each Ψ_j is the subset of formulas amongst $\psi_1, \ldots, \psi_m$ satisfied at position j.

Proposition 2 (MIDL $\Leftrightarrow$ EMITL). *Every MIDL translates to an equivalent EMITL formula, and every EMITL translates to an equivalent MIDL formula.*

We translate an MIDL formula into EMITL by recursively replacing every suffix conjunction $\langle \varphi \rangle \psi$ with the modality $\mathcal{A}(\psi_1, \ldots, \psi_m, \psi)$, such that $\mathcal{A}$ translates the expression $\varphi \cdot \psi ? \cdot (\top ?)^*$ in which atomic expressions $\psi_1 ?, \ldots, \psi_m ?, \psi ?$ are replaced by compatible subsets of $\{\psi_1, \ldots, \psi_m, \psi\}$.

Example 3 (Continued). Formula $\xi \equiv \Box(a \rightarrow (\top ?^* \cdot b \cdot \top ?^+)_{(0,1)} c)$ is rewritten in EMITL as $\xi' \equiv \Box(a \rightarrow \mathcal{B})$, where $\mathcal{B}$ is the MIA given in Figure 6.

Conversely EMITL translate to MIDL replacing automata $\mathcal{A}(\psi_1, \ldots, \psi_m)$ with suffix conjunctions $\langle \varphi \rangle \neg \langle \top ? \rangle \top$, where φ translates $\mathcal{A}$. Here the role of subformula $\neg \langle \top ? \rangle \top$ is to recognize the last position in the word.

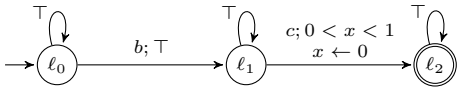

Fig. 6. Automaton $\mathcal{B}$ appearing as subformula in ξ'.

Expressiveness. Supplementing MITL with automata modalities has been proposed by [37] and [17]. The logic of [37] corresponds to the MIDL fragment where all modalities φ are of the form $\langle\gamma\rangle_I$, equivalently, where no star is applied to a timed expression. We call this fragment *basic MIDL*, and show that it is strictly less expressive. In particular, the MIRE $\chi \equiv a \cdot ((a^+)_{[1,2]})^+$ of Example 2 cannot be expressed as a basic MIDL formula. The family (w_n) of Section 3 is not a witness of this fact, since it can be classified for χ using a simple modulo-2 counter. Instead we consider timed words w_n^k, with $k > 0$ events clustered around the events in w_n, as illustrated in Figure 7. Formally, we let $w_0^k = \epsilon$ and $w_n^k = t_n^k\, a\, w_{n-1}^k$ for all $n > 0$, with delays t_n^k given by $t_n^k = \frac{1}{2} + \frac{1}{4k}$ if $n \equiv 0 \pmod k$, $t_n^k = \frac{1}{4k}$ otherwise. We claim that for any basic formula ψ there is a k such that for large enough n either both w_n^k and w_{n+k}^k satisfy ψ, or neither. This disagrees with χ, which recognizes exactly one of w_n^k and w_{n+k}^k.

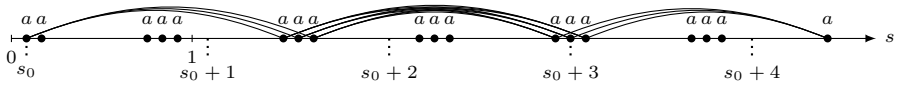

Fig. 7. Timed word w_{18}^3 with events occurring at absolute times $s = s_0, \ldots, s_{17}$. Expression $\chi \equiv a \cdot ((a^+)_{[1,2]})^+$ entails several decompositions of w_{18} as shown. Over words w_n^3 the number of events per interval $[s_i + c, s_i + d]$ for $c < d$ fixed integers and fixed n is either constant or periodic with period 3 as a function of $i < n - 4d$.

6 From MIDL to Nondeterministic Timed Automata

In this section we present a compositional translation of MIDL based on *temporal testers* [27,33,26]. The first step of the procedure turns the MIDL formula into an EMITL formula, and we consider this step implicit. The second step builds testers for every operator of the formula, and composes them together.

Temporal Testers. We introduce the framework of our translation. Let B be a set of Boolean variables. Valuations $u : B \to \{0, 1\}$ are identified with elements of 2^B, under the convention that $u(p) = 1$ iff $p \in u$, for any $p \in B$. In the interest of simplicity, we assume an alphabet of events of the form $\Sigma = 2^A$. We call *timed component* an automaton over an alphabet Σ' of the form 2^B for some $B \supseteq A$. The projection of a timed word $w = t_1 u_1 \ldots t_n u_n$ over variables B onto variables

A is defined as $w|_A = t_1(u_1 \cap A) \dots t_n(u_n \cap A)$. The synchronous product $\mathcal{T}_1 \otimes \mathcal{T}_2$ of timed components $\mathcal{T}_1$ and $\mathcal{T}_2$, defined in the expected way, is such that a timed word w is accepted by $\mathcal{T}_1 \otimes \mathcal{T}_2$ iff w is accepted by both $\mathcal{T}_1$ and $\mathcal{T}_2$ when projected onto their respective variables (see [26] for more details). Let ψ be a formula over $\Sigma = 2^A$ and $\mathcal{T}$ a timed component over $\Sigma' = 2^B$ with output variable $p \in B \setminus A$. We say that $\mathcal{T}[p]$ is a *tester* of ψ when the following conditions hold:

1. For all timed words w over Σ there exists a timed word w' accepted by $\mathcal{T}$ such that w is the projection of w';
2. For all timed words w' accepted by $\mathcal{T}$, and all positions $0 < i \leq |w'|$ it holds $(w', i) \models p$ if and only if $(w', i) \models \psi$.

Compositionality. The construction of a tester $\mathcal{T}_\psi[p]$ for formula ψ is inductive on the structure of ψ. For each subformula ψ' of ψ, we construct a tester for its main subformulas, and compose it with a tester associated to its main operator:

$$\begin{aligned}
\mathcal{T}_{\neg\varphi}[p] &= \mathcal{T}_{\neg q}[p] \otimes \mathcal{T}_\varphi[q] \\
\mathcal{T}_{\varphi \vee \psi}[p] &= \mathcal{T}_{q \vee r}[p] \otimes \mathcal{T}_\varphi[q] \otimes \mathcal{T}_\psi[r] \\
\mathcal{T}_{\mathcal{A}(\psi_1,\dots,\psi_m)}[p] &= \mathcal{T}_{\mathcal{A}(q_1,\dots,q_m)}[p] \otimes \mathcal{T}_{\psi_1}[q_1] \otimes \dots \otimes \mathcal{T}_{\psi_m}[q_m].
\end{aligned}$$

Testers for atomic formulas and propositional operators are simple one-state components, with edges labeled by matching valuations of variables. Testers for automata modalities are presented in the rest of this section. An acceptor $\mathcal{A}_\psi$ of $\mathcal{L}(\psi)$ is obtained by product of $\mathcal{T}_\psi[p]$ with a two-state component enforcing that p holds at position 1 in the input word, and projection onto $\Sigma = 2^A$.

Automata Modalities. For a given MIA $\mathcal{A} = (\Sigma, X, L, S, F, \Delta)$, the tester $\mathcal{T}_\mathcal{A}[p]$ predicts at each position whether $\mathcal{A}$ accepts the corresponding suffix, and outputs the prediction in p. If $\mathcal{T}_\mathcal{A}[p]$ predicts that $\mathcal{A}$ accepts the suffix from i, then it creates a *positive obligation* attached to an initial state, and nondeterministically follows one run of $\mathcal{A}$ from this state. If $\mathcal{T}_\mathcal{A}[p]$ predicts that $\mathcal{A}$ rejects the suffix from i, then it creates a *negative obligation* attached to all initial states, and deterministically follows all runs of $\mathcal{A}$ from those states.

Let c and d be the maximum magnitude and minimum width of clock constraints in $\mathcal{A}$, defined as in Section 4. We define $\preccurlyeq$ as the approximation relation that verifies $C \preccurlyeq D$ when $C \setminus \{(\ell, I), (\ell, J)\} = D \setminus \{(\ell, \mathrm{ch}(I \cup J))\}$ for some distinct $(\ell, I), (\ell, J), (\ell, K) \in C$, $(\ell, \mathrm{ch}(I \cup J)) \in D$ such that $\sup I \cup J \cup K - \inf I \cup J \cup K < d$ and $K \cap \mathrm{ch}(I \cup J) = \emptyset$, or $\inf K > \inf J > c$.

Approximation $\preccurlyeq$ nondeterministically merges two intervals amongst three within the same window of length d. Assume $\inf J \leq \inf K \leq \inf H$; after a delay $t \in \mathbb{T}$ if $t \oplus K \subseteq I$ then either $t \oplus J \subseteq I$ or $t \oplus H \subseteq I$. Similar remarks can be made for intervals above c; this settles the correctness of the approximation relative to positive obligations. For negative obligations we see that $\preccurlyeq$ is finer than $\prec$ of Section 4. The approximation $\prec$ merges intervals separated by a period less than d, while $\preccurlyeq$ merges intervals lying in a window less than d long.

Let m and n be the number of locations, and resolution of $\mathcal{A}$. Any D such that $C \overset{\preccurlyeq}{\dashrightarrow} D$ for some C now has at most $2mn$ interval states.

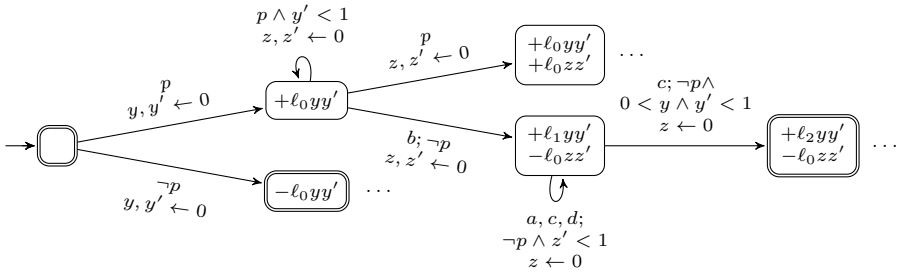

Fig. 8. A few locations and edges of component $\mathcal{T}_{\mathcal{B}}[p]$ (for convenience, the value of p is handled using additional propositional constraints p and $\neg p$).

Subset Construction. We transform the MIA $\mathcal{A} = (\Sigma, X, L, S, F, \Delta)$ into the tester $\mathcal{T}_{\mathcal{A}}[p] = (\Sigma', X', L', S', F', \Delta')$ defined as follows.

- Clocks: $X' = Y \cup Y'$, where $Y = \{y_1, \ldots, y_{2mn}\}$ and $Y' = \{y'_1, \ldots, y'_{2mn}\}$.
- Locations: $L' = 2^{\{-,+\} \times L \times Y \times Y'}$, sets of (negative, positive) obligations.
- Initial locations: $E' = \{\emptyset\}$.
- Accepting locations: $F' = 2^{(\{-\} \times (L \setminus F) \cup \{+\} \times F) \times Y \times Y'}$, such that all positive (negative) obligations are attached to accepting (rejecting) states.
- Edges: we define $\lambda_{\preccurlyeq}(R, Q)$, Z_Q and $\overline{Q}$ similarly as in Section 4, and let

$$\Delta' = \{(P, u, \lambda_{\preccurlyeq}(R, Q) \wedge \theta(P, E), Z_Q, \overline{Q}) \mid P \in L', u \in \Sigma', E \in \Delta(P, u \cap \Sigma), \\ R \in R_{u(p)}(P, E), Q \in L'_0\}$$

  where $\theta(P, E)$, $\Delta(P, a)$ for $a \in \Sigma$, and $R_i(P, E)$ for $i = 0, 1$ are defined below.

Given $P \in L'$ and $a \in \Sigma$, we denote $\Delta(P, a)$ the set of subsets $E \subseteq \Delta$ such that for all $+\ell x x' \in P$ there exists $\delta = (\ell, a, \phi, Z, \ell') \in E$, and for all $-\ell x x' \in P$ and $\delta = (\ell, a, \phi, Z, \ell')$, if $\delta \in \Delta$ then $\delta \in E$. The constraint $\theta(P, E)$, defined similarly as in Section 4, ensures E contains all feasible edges from negatively marked locations, and one feasible edge from each positively marked location. We denote L'_0 locations of L' with additional variables y_0, y'_0 as previously. Define the target configuration $R(P, E) \in L'_0$ when taking edges E from P as follows: $R(P, E) = \{s\ell' yy' \mid (\ell, a, \top, \emptyset, \ell') \in E, s\ell yy' \in P\} \cup \{s\ell' y_0 y'_0 \mid (\ell, a, \phi, \{x\}, \ell') \in E, s\ell yy' \in P\}$. When the prediction p is false the set of possible target configurations is given by $R_0(P, E) = \{R(P, E)\}$, and when the prediction p is true, given by $R_1(P, E) = \{(R(P, E) \cup \{+\ell y_0 y'_0\}) \mid \ell \in S\}$. This completes the construction of $\mathcal{T}_{\mathcal{A}}[p]$.

Theorem 4 (MIDL $\Rightarrow$ NTA). *For any MIDL formula of size m and resolution n one can construct an equivalent timed automaton $\mathcal{A}_\varphi$ with $4mn$ clocks and $2^{8m^3n^2+1}$ locations.*

Example 3 (Continued). Consider automaton $\mathcal{B}$ of Figure 6. We illustrate the process of constructing its tester in Figure 8. After constructing $\mathcal{T}_{\mathcal{B}}[p]$, we obtain the tester for ξ' as the product $\mathcal{T}_{\xi'}[r] = \mathcal{T}_{\Box q}[r] \otimes \mathcal{T}_{a \to p}[q] \otimes \mathcal{T}_{\mathcal{B}}[p]$. To get an acceptor for ξ, we take the product of $\mathcal{T}_{\xi'}[r]$ with some acceptor of r and project back the result onto alphabet Σ.

7 Discussion

We extended the punctuality relaxation of [3] to timed versions of regular expressions and dynamic logic, generalizing results of [37,17]. The expressions we introduced have a direct connection with automata. Their expressiveness is limited to a small class of one-clock timed automata, also related to *perturbed timed automata* [5]. However in the setting of dynamic logic, such expressions yield an expressive specification language with good decidability properties:

Corollary 1. *The satisfiability of MIDL and the model checking of timed automata against MIDL are EXPSPACE-complete.*

The lower bound follows from the discrete-time case, while the upper bound is obtained by reduction to timed automata language emptiness, see [3]. Decision procedures for MITL have recently been gaining interest, with implementations of [8,12] and formalization by [34]. An interesting direction for future work would be to assess experimentally the efficiency of MIDL decision procedures derived from Corollary 1.

Metric dynamic logic was independently proposed by [7] in the context of monitoring. Extensions of metric temporal logic with regular expressions modalities were also studied by [23]. The logic *MITL+URat* of [23] is equivalent to *basic MIDL* discussed in the present paper. Its modalities $\psi_1 \mathcal{U}_{\gamma,I} \psi_2$ can be written $\langle \gamma \cap (\psi_1?^* \cdot \top?) \rangle_I \psi_2$ (the intersection $\cap$ of untimed expressions γ and $\psi_1?^* \cdot \top?$ can be eliminated in polynomial time) and in the other direction $\langle \gamma \rangle_I \psi$ rewrites into $\top \mathcal{U}_{\gamma,I} \psi$. Both logics are equivalent (and translate in polynomial time) to the *EMITL* of [37]. Complexity of the satisfiability problem was not studied by [37], whose proofs can only give non-elementary upper bounds. The present work improves on the 2EXPSPACE upper bound of [23] by providing a tight EXPSPACE construction. The more general *MITL+Rat* [23] has non-elementary complexity. The position of MIDL in the expressiveness landscape of decidable MTL variants (see also [22]) is a topic for future research.

The family of languages considered in this paper are all recognizable by one-clock *alternating* timed automata (OCATA) [24,30]. Our determinization procedure uses a timed variant of the classical subset construction inspired from [10]. The authors of [10,11] consider the dual problem of eliminating universal non-determinism in OCATA stemming from the translation [30] of MITL formulas. The transition graph in an MIA has more structure than in OCATA stemming from MITL translations, requiring additional clocks to follow states moving to the same location using separate paths. While the emptiness problem for OCATA is decidable over finite words, its complexity is non-elementary [24,30]. Generalizations of this model with Büchi conditions, two-wayness, or silent transitions all lead to undecidability [1]. On the contrary our expressions and logic have elementary decision procedures, which can in principle be extended to handle ω-words, past operators, and continuous-time Boolean signals.

Acknowledgments. I thank Eugene Asarin, Tom Henzinger, Oded Maler, Dejan Ničković, and anonymous reviewers of multiple conferences for their helpful feedback.

References

1. Parosh Aziz Abdulla, Johann Deneux, Joël Ouaknine, Karin Quaas, and James Worrell. Universality analysis for one-clock timed automata. *Fundam. Inform.*, 89(4):419–450, 2008.
2. Rajeev Alur and David L Dill. A theory of timed automata. *Theoretical computer science*, 126(2):183–235, 1994.
3. Rajeev Alur, Tomás Feder, and Thomas A Henzinger. The benefits of relaxing punctuality. *Journal of the ACM*, 43(1):116–146, 1996.
4. Rajeev Alur and Thomas A Henzinger. Logics and models of real time: A survey. In *Workshop/School/Symposium of the REX Project*, pages 74–106. Springer, 1991.
5. Rajeev Alur, Salvatore La Torre, and Parthasarathy Madhusudan. Perturbed timed automata. In *Hybrid Systems: Computation and Control*, pages 70–85. Springer, 2005.
6. Eugene Asarin, Paul Caspi, and Oded Maler. Timed regular expressions. *Journal of the ACM*, 49(2):172–206, 2002.
7. David Basin, Srđan Krstić, and Dmitriy Traytel. Almost event-rate independent monitoring of metric dynamic logic. In *Runtime Verification*, pages 85–102. Springer, 2017.
8. Marcello M Bersani, Matteo Rossi, and Pierluigi San Pietro. A tool for deciding the satisfiability of continuous-time metric temporal logic. *Acta Informatica*, 53(2):171–206, 2016.
9. Patricia Bouyer, Fabrice Chevalier, and Nicolas Markey. On the expressiveness of TPTL and MTL. In *Foundations of Software Technology and Theoretical Computer Science*, pages 432–443. Springer, 2005.
10. Thomas Brihaye, Morgane Estiévenart, and Gilles Geeraerts. On MITL and alternating timed automata. In *Formal Modeling and Analysis of Timed Systems*, pages 47–61, 2013.
11. Thomas Brihaye, Morgane Estiévenart, and Gilles Geeraerts. On MITL and alternating timed automata over infinite words. In *Formal Modeling and Analysis of Timed Systems*, pages 69–84, 2014.
12. Thomas Brihaye, Gilles Geeraerts, Hsi-Ming Ho, and Benjamin Monmege. Mightyl: A compositional translation from MITL to timed automata. In *Computer Aided Verification*, pages 421–440, 2017.
13. Giuseppe De Giacomo and Moshe Y Vardi. Linear temporal logic and linear dynamic logic on finite traces. In *IJCAI*, volume 13, pages 854–860, 2013.
14. Cindy Eisner and Dana Fisman. *A Practical Introduction to PSL (Series on Integrated Circuits and Systems)*. Springer-Verlag, 2006.
15. Michael J Fischer and Richard E Ladner. Propositional dynamic logic of regular programs. *Journal of computer and system sciences*, 18(2):194–211, 1979.
16. Carlo A. Furia and Matteo Rossi. MTL with bounded variability: Decidability and complexity. In *Formal Modeling and Analysis of Timed Systems*, pages 109–123. Springer-Verlag, 2008.
17. Thomas A Henzinger, J-F Raskin, and P-Y Schobbens. The regular real-time languages. In *Automata, Languages and Programming*, pages 580–591. Springer, 1998.
18. Yoram Hirshfeld and Alexander Rabinovich. An expressive temporal logic for real time. In *Mathematical Foundations of Computer Science 2006*, pages 492–504. Springer, 2006.

19. Yoram Hirshfeld and Alexander Rabinovich. Expressiveness of metric modalities for continuous time. In *Computer Science–Theory and Applications*, pages 211–220. Springer, 2006.
20. Stephen Cole Kleene. Representation of events in nerve nets and finite automata. *Automata Studies*, pages 3–42, 1956.
21. Ron Koymans. Specifying real-time properties with metric temporal logic. *Real-time systems*, 2(4):255–299, 1990.
22. Shankara Narayanan Krishna, Khushraj Madnani, and Paritosh K Pandya. Metric temporal logic with counting. In *Foundations of Software Science and Computation Structures*, pages 335–352. Springer, 2016.
23. Shankara Narayanan Krishna, Khushraj Madnani, and Paritosh K Pandya. Making metric temporal logic rational. In *Mathematical Foundations of Computer Science*, pages 77:1–77:14, 2017.
24. Sławomir Lasota and Igor Walukiewicz. Alternating timed automata. In *Foundations of Software Science and Computation Structures*, pages 250–265. Springer, 2005.
25. Oded Maler, Dejan Nickovic, and Amir Pnueli. Real time temporal logic: Past, present, future. In *Formal Modeling and Analysis of Timed Systems*, pages 2–16. Springer, 2005.
26. Oded Maler, Dejan Nickovic, and Amir Pnueli. From MITL to timed automata. In *Formal Modeling and Analysis of Timed Systems*, pages 274–289. Springer, 2006.
27. Max Michel. Composition of temporal operators. *Logique et Analyse*, 28(110/111):137–152, 1985.
28. Dejan Ničković and Nir Piterman. From MTL to deterministic timed automata. In *Formal Modeling and Analysis of Timed Systems*, pages 152–167. Springer, 2010.
29. Joël Ouaknine, Alexander Rabinovich, and James Worrell. Time-bounded verification. In *Concurrency Theory*, pages 496–510. Springer, 2009.
30. Joël Ouaknine and James Worrell. On the decidability of metric temporal logic. In *Logic in Computer Science*, pages 188–197. IEEE, 2005.
31. Joël Ouaknine and James Worrell. On metric temporal logic and faulty turing machines. In *Foundations of Software Science and Computation Structures*, pages 217–230. Springer, 2006.
32. Amir Pnueli. The temporal logic of programs. In *Foundations of Computer Science*, pages 46–57. IEEE, 1977.
33. Amir Pnueli and Aleksandr Zaks. On the merits of temporal testers. In *25 Years of Model Checking*, pages 172–195. Springer, 2008.
34. Nima Roohi and Mahesh Viswanathan. Revisiting MITL to fix decision procedures. In *Verification, Model Checking, and Abstract Interpretation*, pages 474–494. Springer, 2018.
35. Michael Sipser. *Introduction to the Theory of Computation*, volume 2. Thomson Course Technology Boston, 2006.
36. Moshe Y. Vardi. From philosophical to industrial logics. In *ICLA*, volume 5378 of *Lecture Notes in Computer Science*, pages 89–115, 2009.
37. Thomas Wilke. Specifying timed state sequences in powerful decidable logics and timed automata. In *Formal Techniques in Real-Time and Fault-Tolerant Systems*, pages 694–715. Springer, 1994.