Discrete Meets Continuous, Again

Carlo Alberto Furia

December 2006

Abstract

This report collects some contributions about issues related to the notion of *sampling invariance* that has been introduced in [FR05, FR06], of which the present report can therefore be considered a follow-up and a complement.

In particular: we compare the expressiveness of the language $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO against that of MTL (and MITL), by also showing a result about the expressiveness of non-strict temporal operators; we characterize behaviors of bounded variability and formulas that preserve the bounded variability requirement; we compare the notion of sampling invariance with that of digitization [HMP92]; we discuss how to describe the runs of a timed automata with $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formulas; we provide an example of application of the sampling invariance techniques to a simple verification problem; we summarize several related works on the expressiveness, decidability, and complexity of formalisms that are somewhat related to the $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO temporal logic.

Contents

This report collects some contributions about issues gravitating around the notion of *sampling invariance* that we introduced in [FR05, FR06]. More precisely, here it is a summary of the various sections.

- Section 1 compares the expressiveness of the $^{\mathbb{R}}_{\mathbb{Z}}$TRIO[1] language against that of other related formalisms, and most notably MTL and MITL. This allows one to realize to what extent the results about the sampling invariance of $^{\mathbb{R}}_{\mathbb{Z}}$TRIO carry over to other similar formalisms, and *vice versa* how to apply well-known results about expressiveness and decidability to the $^{\mathbb{R}}_{\mathbb{Z}}$TRIO language. In particular, Section 1.1 contributes by showing the equivalence of strict and non-strict operators over continuous time, contrarily to what it is generally held.

- Section 2 tries to characterize (and give a name!) to the behaviors subject to the regularity constraint χ (sometimes also called of *bounded variability* [Wil94]). More precisely, a class of $^{\mathbb{R}}_{\mathbb{Z}}$TRIO formulas that preserve the regularity requirement is shown; such formulas can therefore be nested without compromising sampling invariance of a formula. Then, a mathematical (partial) characterization of regular behaviors over dense-valued items is given and discussed.

- Section 3 compares the notion of sampling invariance with another well-known notion of discretization, namely *digitization* [HMP92]. It is shown that, according to a given comparison framework, the two notions are orthogonal. Other aspects — that may be the object of other, different comparisons — are also briefly discussed.

- Section 4 discusses how timed automata [AH92a] can be formalized in $^{\mathbb{R}}_{\mathbb{Z}}$TRIO. This allows one to reason about timed automata runs using temporal logic, thus pursuing a sort of dual-language approach.

- Section 5 provides a fully-discussed example of usage of the sampling invariance framework to the verification of a simple controlled system, specified in TRIO. The same example has been presented in [FR06].

- Finally, Section 6 presents summaries of several (more or less recent) works about the expressiveness, decidability, and complexity of formalisms which are close to $^{\mathbb{R}}_{\mathbb{Z}}$TRIO in expressive power. These papers also show the several heterogeneous semantic models that have been introduced in the field, and relate them in a formal way.

1 On the Expressiveness of $^{\mathbb{R}}_{\mathbb{Z}}$TRIO

This section compares the expressiveness of $^{\mathbb{R}}_{\mathbb{Z}}$TRIO with those of other related formalisms, and discusses how the nesting-freeness requirement on formulas also influences the expressiveness of the resulting language.

[1] Notice that $^{\mathbb{R}}_{\mathbb{Z}}$TRIO was called TRIO_{si} in [FR05].

1.1 Strict vs. Non-Strict Operators

$\frac{\mathbb{R}}{\mathbb{Z}}$TRIO definition of the *until* and *since* operators is *non-strict* in the first argument, in that the first argument is required (in particular) to hold at the current instant as well (see [FR05]). This choice is not very common, and it has been made necessary to achieve sampling invariance. On the contrary, the usual choice for dense-time domains is a *strict* one. This section discusses how the choice of strict and non-strict operators impacts the expressiveness of a language, by showing that non-strict metric operators are as expressive as strict ones.

It is usually held that the expressive equivalence between strict and non-strict operators does not hold, in general, over dense time, where strict variants are believed to be strictly more expressive. This is a "folk theorem" which has never been explicitly proved — to the best of our knowledge — but whose validity is generally accepted. For instance, [BCM05] claims that the formula $\neg a \wedge \widetilde{\text{Until}}_{(0,+\infty)}(a, b)$ — where $\widetilde{\text{Until}}$ denotes a strict variation of the until operator that we are going to define shortly — cannot be expressed using only non-strict until.[2] [Hen98] claims in Footnote 5 that "in real time, strict until cannot be defined from weak until and next"[3] citing a personal communication with Raskin as a reference. [AFH96] claims that MITL's *bounded until* "cannot be defined in terms of an *until* operator that is not strict in its first argument".

However, these claim are not true for our continuous-time semantics, if we restrict ourselves to non-Zeno behaviors (as it is customary, and as it is implicit with models such as the interval-based sequence), as we are now going to show. Let us remark that we are focusing on the dense-time semantics, unless explicitly stated otherwise.

Strict semantics of operators. Let us define formally the semantics of strict *until* and *since*, denoted as $\widetilde{\text{Until}}$ and $\widetilde{\text{Since}}$, respectively.

$b(t) \models_{\mathbb{T}} \widetilde{\text{Until}}_{I\rangle}(\phi_1, \phi_2)$ iff there exists $d \in I$ such that $b(t+d) \models_{\mathbb{T}} \phi_2$ and, for all $u \in (0, d\rangle$ it is $b(t+u) \models_{\mathbb{T}} \phi_1$

$b(t) \models_{\mathbb{T}} \widetilde{\text{Since}}_{I\langle}(\phi_1, \phi_2)$ iff there exists $d \in I$ such that $b(t-d) \models_{\mathbb{T}} \phi_2$ and, for all $u \in \langle -d, 0)$ it is $b(t+u) \models_{\mathbb{T}} \phi_1$

We refer to the definitions in [FR05] for the semantics of the non-strict *until* and *since*, as well as the derived operators that we are going to use. In particular, let us remark that we always assume that the intervals I are non empty, i.e., the right end-point is at least as large as the left end-point. It is not difficult to see that this is without loss of generality.

Strict is at least as expressive as non-strict. It is quite clear that strict metric operators are at least as expressive as non-strict one. In fact, in general

[2] Actually, it is not clear if [BCM05] refers the claim to qualitative and/or discrete-time temporal logic.

[3] Notice, however, that the reference is to a point-based semantics, while we consider interval-based models.

the following equivalences hold both over dense and over discrete time.

$$\mathrm{Until}_{I\rangle}(\phi_1,\phi_2) \equiv \begin{cases} \phi_2 \vee (\phi_1 \wedge \widetilde{\mathrm{Until}}_{(0,u\rangle)}(\phi_1,\phi_2)) & \text{if } I=[0,u\rangle \text{ and } \rangle \text{ is)} \\ \phi_1 \wedge \widetilde{\mathrm{Until}}_{I\rangle}(\phi_1,\phi_2) & \text{otherwise, i.e., if } 0 \notin I \text{ or } \rangle \text{ is]} \end{cases}$$

The case for the *since* is immediately derivable.

Over discrete time, strict is exactly as expressive as non-strict. Over *discrete time*, it is also straightforward to notice that the converse holds, i.e., non-strict operators are at least as expressive as strict ones. In fact we have the following (recall that, in discrete time, all intervals can be expressed as closed ones).

$$\widetilde{\mathrm{Until}}_{[l,u]\rangle}(\phi_1,\phi_2) \equiv^{\mathbb{Z}} \begin{cases} \mathrm{NowOn}\big(\mathrm{Until}_{[l-1,u-1]\rangle}(\phi_1,\phi_2)\big) & \text{if } l \geq 1 \\ \mathrm{NowOn}\big(\mathrm{Until}_{[l,u-1]\rangle}(\phi_1,\phi_2)\big) \vee \phi_2 & \text{if } l=0<u \\ \phi_2 & \text{if } l=u=0 \end{cases}$$

The case for the *since* is immediately derivable.

Notice that, in the above equivalences, we used the NowOn operator, which in discrete time can be expressed as $\mathrm{NowOn}(\phi) = \mathrm{Futr}(\phi,1)$, which is the same as the usual Linear Temporal Logic (LTL) *next* operator [GHR94]. Therefore, the equivalence in expressive power over discrete time does not contradict the well-known result that *next* in LTL cannot be defined from non-strict *until* only [GHR94], as the result for LTL does not deal with the metric modality of *bounded until* — which is instead the basis of $^{\mathbb{R}}_{\mathbb{Z}}$TRIO — but with qualitative (unbounded) *until* only.

Strict and non-strict NowOn operators. In order to better comprehend the problem in the dense-time setting, let us strip it down to its core. Let us define two variations of the *nowon* and *uptonow* operators which are useful over dense time. $\epsilon\mathrm{NowOn}(\phi)$ denotes that ϕ holds in a non-empty right-open left-closed interval on the right of the current instant; therefore $\epsilon\mathrm{NowOn}(\phi) \equiv \mathrm{Until}_{(0,+\infty))}(\phi,\mathrm{true})$. Its past counterpart is $\epsilon\mathrm{UpToNow}(\phi)$; it denotes that ϕ holds in a non-empty left-open right-closed interval on the left of the current instant, and it is defined as $\epsilon\mathrm{UpToNow}(\phi) \equiv \mathrm{Since}_{(0,+\infty)(}(\phi,\mathrm{true})$. Their *strict* counterparts predicate over intervals that are open on both ends; their definitions are the following: $\epsilon\widetilde{\mathrm{NowOn}}(\phi) \equiv \widetilde{\mathrm{Until}}_{(0,+\infty))}(\phi,\mathrm{true})$ and $\epsilon\widetilde{\mathrm{UpToNow}}(\phi) \equiv \widetilde{\mathrm{Since}}_{(0,+\infty)(}(\phi,\mathrm{true})$.[4]

As we are going to show more explicitly shortly, the problem of expressiveness for the strict vs. non-strict *until* and *since* requires in particular to express the strict *nowon* and *uptonow* using their non-strict counterparts. In order to

[4] Recall that in standard TRIO the operators $\epsilon\widetilde{\mathrm{NowOn}}$ and $\epsilon\widetilde{\mathrm{UpToNow}}$ are denoted simply as NowOn and UpToNow [GM01]. Here, however, we adopt the notation and definitions of $^{\mathbb{R}}_{\mathbb{Z}}$TRIO [FR05, FR06].

achieve this, we have to first recall a basic property of non-Zeno time-dependent predicates. As it is shown in [GM01], any non-Zeno primitive p item obeys the following formulas:

$$\mathrm{Alw}\Big(\epsilon\widetilde{\mathrm{NowOn}}(\mathsf{p}) \vee \epsilon\widetilde{\mathrm{NowOn}}(\neg\mathsf{p})\Big)$$
$$\mathrm{Alw}\Big(\epsilon\widetilde{\mathrm{UpToNow}}(\mathsf{p}) \vee \epsilon\widetilde{\mathrm{UpToNow}}(\neg\mathsf{p})\Big)$$

That is, there is always a non-empty interval in the future (and one in the past) where p has a definite constant value. It is not difficult to "lift" such property from basic items to arbitrary formulas, by showing that for any ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO formula ϕ, if all of the basic items on which it predicates are non-Zeno, then the truth value of ϕ as a function of time is also non-Zeno. In other words, (temporal) operators preserve non-Zenoness. The proof would go by structural induction. So we have, for any ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO formula ϕ, that:

$$\mathrm{Alw}\Big(\epsilon\widetilde{\mathrm{NowOn}}(\phi) \vee \epsilon\widetilde{\mathrm{NowOn}}(\neg\phi)\Big) \quad (1)$$
$$\mathrm{Alw}\Big(\epsilon\widetilde{\mathrm{UpToNow}}(\phi) \vee \epsilon\widetilde{\mathrm{UpToNow}}(\neg\phi)\Big) \quad (2)$$

Therefore, we exploit this basic non-Zenoness property to express strict operators using non-strict ones. We have the following equivalences.

$$\epsilon\widetilde{\mathrm{NowOn}}(\phi) \;\Leftrightarrow\; \epsilon\mathrm{NowOn}(\phi) \vee (\neg\phi \wedge \neg\, \epsilon\mathrm{NowOn}(\neg\phi))$$
$$\epsilon\widetilde{\mathrm{UpToNow}}(\phi) \;\Leftrightarrow\; \epsilon\mathrm{UpToNow}(\phi) \vee (\neg\phi \wedge \neg\, \epsilon\mathrm{UpToNow}(\neg\phi))$$

The proof is simple, and relies on properties 1 and 2. Let us demonstrate the proof for the *nowon* case, the other being obviously all similar.

Proof. Let us start by proving the $\Rightarrow$ direction: assume that $\epsilon\widetilde{\mathrm{NowOn}}(\phi)$ holds at some instant t. Let us first consider the case: ϕ true at t; then also $\epsilon\mathrm{NowOn}(\phi)$, and we satisfy the first term of the disjunction. Otherwise, let us consider ϕ false at t. Then, $\epsilon\mathrm{NowOn}(\neg\phi)$ must also be false at t, otherwise $\epsilon\widetilde{\mathrm{NowOn}}(\phi)$ cannot be true. This concludes this branch, since we satisfy the second term of the disjunction.

For the $\Leftarrow$ direction, let us start by considering the case $\epsilon\mathrm{NowOn}(\phi)$ at t; then, *a fortiori*, $\epsilon\widetilde{\mathrm{NowOn}}(\phi)$, at t and we are done. Otherwise, let us assume that $\neg\phi \wedge \neg\, \epsilon\mathrm{NowOn}(\neg\phi)$ holds at the current instant t. By evaluating Formula 1 at time t, we immediately conclude that $\epsilon\widetilde{\mathrm{NowOn}}(\phi)$ at t, which concludes the whole proof. □

Strict until expressed with non-strict operators. Let us assume $a > 0$; then, the following equivalence hold.

$$\widetilde{\mathrm{Until}}_{(a,b\rangle)}(\phi_1, \phi_2) \;\Leftrightarrow\; \mathrm{Until}_{(a,b\rangle)}(\mathrm{true}, \phi_2) \wedge \mathrm{Lasts}_{\mathrm{ei}}\big(\mathrm{Until}_{(0,+\infty))}(\phi_1, \phi_2)\,, a\big) \quad (3)$$

Proof of Formula (3). Let us start with the $\Rightarrow$ direction: assume that $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{(a,b\rangle)}(\phi_1, \phi_2)$. That is, there exists a $u \in (t+a, t+b\rangle$ such that $b(u) \models_{\mathbb{R}} \phi_2$ and, for all $v \in (t,u)$ it is $b(v) \models_{\mathbb{R}} \phi_1$. From $b(u) \models_{\mathbb{R}} \phi_2$ it follows immediately that $\text{Until}_{(a,b\rangle)}(\text{true}, \phi_2)$, so the first conjunct is proved.

Then, let us show that $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ei}}\big(\text{Until}_{(0,+\infty))}(\phi_1, \phi_2)\,, a\big)$. Let α be any instant in $(0, a]$; we have to show that $b(t+\alpha) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \phi_2)$. Notice that $(t+a, t+b\rangle \subseteq (t+\alpha, +\infty)$, as $t+a \geq t+\alpha$, therefore $u \in (t+\alpha, +\infty)$ *a fortiori*. Moreover, $[t+\alpha, u) \subset (t,u)$, as $\alpha > 0$, so for all $v' \in [t+\alpha, u)$ it is $b(v') \models_{\mathbb{R}} \phi_1$. Therefore, we have shown that $b(t+\alpha) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \phi_2)$.

Let us now consider the $\Leftarrow$ direction. First of all, let us notice that $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ei}}\big(\text{Until}_{(0,+\infty))}(\phi_1, \phi_2)\,, a\big)$ implies that $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ee}}(\phi_1, a)$. Moreover, in particular $b(t+a) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \phi_2)$. That is, there exists a $u \in (t+a, +\infty)$ such that $b(u) \models_{\mathbb{R}} \phi_2$ and, for all $v \in [t+a, u)$ it is $b(v) \models_{\mathbb{R}} \phi_1$.

Let us now consider the case $u \in (t+a, t+b\rangle$. All in all, ϕ_1 holds over the interval (t,u), and ϕ_2 holds at u; therefore, we have $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{(a,b\rangle)}(\phi_1, \phi_2)$.

Otherwise, let us consider the case $u \notin (t+a, t+b\rangle$; therefore $u \in \langle' t+b, +\infty)$, where $\langle'$ is the "complement" of $\rangle$.[5] In particular, this implies that for all $v \in (t, t+b)$ it is $b(v) \models_{\mathbb{R}} \phi_1$. Moreover, we are also assuming that $b(t) \models_{\mathbb{R}} \text{Until}_{(a,b\rangle)}(\text{true}, \phi_2)$ in this branch of the proof. That is, there exists a $u' \in (t+a, t+b\rangle$ such that $b(u') \models_{\mathbb{R}} \phi_2$. Since $(t,u') \subseteq (t, t+b)$, then we have shown that $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{(a,b\rangle)}(\phi_1, \phi_2)$, as required. □

Now, using Formula 3, it is not difficult to prove the following (still for $a > 0$).

$$\widetilde{\text{Until}}_{[a,b\rangle)}(\phi_1, \phi_2) \;\Leftrightarrow\; \widetilde{\text{Until}}_{(a,b\rangle)}(\phi_1, \phi_2) \vee (\text{Lasts}_{\text{ee}}(\phi_1, a) \wedge \text{Futr}(\phi_2, a)) \quad (4)$$

Proof of Formula (4). Beginning with the $\Rightarrow$ direction, assume that there exists a $u \in [t+a, t+b\rangle$ such that $b(u) \models_{\mathbb{R}} \phi_2$ and for all $v \in (t,u)$ it is $b(v) \models_{\mathbb{R}} \phi_1$. Then, if $u \in (t+a, t+b\rangle$, obviously $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{(a,b\rangle)}(\phi_1, \phi_2)$. Otherwise, it is $u = t+a$; therefore $b(t) \models_{\mathbb{R}} \text{Futr}(\phi_2, a)$ and $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ee}}(\phi_1, a)$.

For the $\Leftarrow$ direction, let us first consider $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{(a,b\rangle)}(\phi_1, \phi_2)$; then there exists a $u \in (t+a, t+b\rangle)$ such that $b(u) \models_{\mathbb{R}} \phi_2$ and for all $v \in (t,u)$ it is $b(v) \models_{\mathbb{R}} \phi_1$. Since $(t+a, t+b\rangle \subset [t+a, t+b\rangle$, then *a fortiori* $b(t)\widetilde{\text{Until}}_{[a,b\rangle)}(\phi_1, \phi_2)$. Otherwise, $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ee}}(\phi_1, a) \wedge \text{Futr}(\phi_2, a)$ implies $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{[a,b\rangle)}(\phi_1, \phi_2)$ for $u = t+a$. □

Variants with]. It is simple to express the] variants of the *until* using the) variants used above; in fact, we have the following.

$$\widetilde{\text{Until}}_{(a,b\rangle]}(\phi_1, \phi_2) \;\Leftrightarrow\; \widetilde{\text{Until}}_{(a,b\rangle)}(\phi_1, \phi_2 \wedge \phi_1) \quad (5)$$

$$\widetilde{\text{Until}}_{[a,b\rangle]}(\phi_1, \phi_2) \;\Leftrightarrow\; \widetilde{\text{Until}}_{[a,b\rangle)}(\phi_1, \phi_2 \wedge \phi_1) \quad (6)$$

[5] That is $\rangle'$ is [and $]'$ is (.

No need for punctual intervals. Notice that Formula 4 uses a *punctual* interval, i.e., it specifies an exact time distance through the Futr operator. Nonetheless, this is not necessary, as far as the expression of the strict *until* is concerned. In fact, we provide alternative equivalent formulas that do not use punctual intervals (of course, provided the strict until itself is constrained by a non-punctual interval). They will be useful in defining the relation between $^{\mathbb{R}}_{\mathbb{Z}}$TRIO and MITL in Section 1.2.3.

This time, we start with the] variant of the *until* operator, and we first establish the following (for $a > 0$).

$$\widetilde{\text{Until}}_{[a,b\rangle]}(\phi_1, \phi_2) \;\Leftrightarrow\; \text{Lasts}_{\text{ei}}\big(\text{Until}_{[0,+\infty)]}(\phi_1, \phi_2), a\big) \wedge \text{Until}_{[a,b\rangle]}(\text{true}, \phi_2 \wedge \phi_1) \quad (7)$$

Proof of Formula 7. Let us start with the $\Rightarrow$ direction, and let t be the current instant. We assume that there exists a $u \in [t+a, t+b\rangle$ such that $b(u) \models_{\mathbb{R}} \phi_2$ and, for all $v \in (t, u]$ it is $b(v) \models_{\mathbb{R}} \phi_1$. Clearly, $b(t) \models_{\mathbb{R}} \text{Until}_{[a,b)]}(\text{true}, \phi_2 \wedge \phi_1)$ is immediately implied. We still have to show that, for all $d \in (t, t+a]$, it is $b(d) \models_{\mathbb{R}} \text{Until}_{[0,+\infty)]}(\phi_1, \phi_2)$.

So, let us consider a generic d; notice that $u \in [d, +\infty)$ as $d \leq t + a \leq u$, and recall that $b(u) \models_{\mathbb{R}} \phi_2$. Moreover, let v' be any point in $[d, u]$; since $d > t$, *a fortiori* $v' \in (t, u]$. Thus, by hypothesis $b(v') \models_{\mathbb{R}} \phi_1$, so we are done with proving $b(d) \models_{\mathbb{R}} \text{Until}_{[0,+\infty)]}(\phi_1, \phi_2)$.

Let us now consider the $\Leftarrow$ direction. First of all, let us realize that $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ei}}\big(\text{Until}_{[0,+\infty)]}(\phi_1, \phi_2), a\big)$ implies $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ee}}(\phi_1, a)$. In fact, otherwise there would be a $y \in (t, t+a)$ such that $b(y) \models_{\mathbb{R}} \neg\phi_1$; but since it is also $b(y) \models_{\mathbb{R}} \text{Until}_{[0,+\infty)]}(\phi_1, \phi_2)$ we clearly have a contradiction.

Next, let us consider the consequences of $b(t+a) \models_{\mathbb{R}} \text{Until}_{[0,+\infty)]}(\phi_1, \phi_2)$: it means that there exists a $u' \in [t+a, +\infty)$ such that $b(u') \models_{\mathbb{R}} \phi_2$ and for all $v' \in [t+a, u']$ it is $b(v') \models_{\mathbb{R}} \phi_1$.

Let us first distinguish the case $u' \in [t+a, t+b\rangle$. All in all, ϕ_1 holds over the interval $(t, u']$, and ϕ_2 holds at u'; therefore, we have $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{[a,b\rangle]}(\phi_1, \phi_2)$.

Otherwise, let us consider the case $u' \notin [t+a, t+b\rangle$; therefore $u' \in \langle' t+b, +\infty)$, where $\langle'$ is the "complement" of $\rangle$. In particular, this implies that for all $v' \in (t, t+b)$ it is $b(v') \models_{\mathbb{R}} \phi_1$. Moreover, we are also assuming that $b(t) \models_{\mathbb{R}} \text{Until}_{[a,b)]}(\text{true}, \phi_2 \wedge \phi_1)$ in this branch of the proof. That is, there exists a $u'' \in [t+a, t+b\rangle$ such that $b(u'') \models_{\mathbb{R}} \phi_2 \wedge \phi_1$. Since $(t, u'') \subseteq (t, t+b)$, then we have shown that $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{[a,b\rangle]}(\phi_1, \phi_2)$, as required. □

Finally, in the following we express the) variant through the] variant (as always, assume $a > 0$).

$$\widetilde{\text{Until}}_{[a,b\rangle)}(\phi_1, \phi_2) \;\Leftrightarrow\; \widetilde{\text{Until}}_{[a,b\rangle]}(\phi_1, \phi_2) \;\vee\; \big(\text{Lasts}_{\text{ee}}\big(\text{Until}_{(0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2), a\big) \wedge \text{Until}_{[a,b\rangle)}(\text{true}, \neg\phi_1 \wedge \phi_2)\big) \quad (8)$$

Proof of Formula 8. Let us start with the $\Rightarrow$ direction, and let t be the current instant. We assume that there exists a $u \in [t+a, t+b\rangle$ such that $b(u) \models_{\mathbb{R}} \phi_2$

and, for all $v \in (t,u)$ it is $b(v) \models_{\mathbb{R}} \phi_1$. If, furthermore, $b(u) \models_{\mathbb{R}} \phi_1$, then $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{[a,b\rangle]}(\phi_1,\phi_2)$ and we are done. Otherwise, let us consider the case $b(u) \models_{\mathbb{R}} \neg\phi_1$: clearly, $b(t) \models_{\mathbb{R}} \text{Until}_{[a,b\rangle]}(\text{true}, \neg\phi_1 \wedge \phi_2)$ is immediately implied (as $\neg\phi_1 \wedge \phi_2$ holds at u). We still have to show that, for all $d \in (t, t+a)$, it is $b(d) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2)$.

So, let us consider a generic d; notice that $u \in (d, +\infty)$ as $d < t + a \leq u$, and recall that $b(u) \models_{\mathbb{R}} \phi_2 \wedge \neg\phi_1$. Moreover, let v' be any point in $[d, u)$; since $d > t$, *a fortiori* $v' \in (t, u)$. Thus, by hypothesis $b(v') \models_{\mathbb{R}} \phi_1$, so we are done with proving $b(d) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2)$.

Let us now consider the $\Leftarrow$ direction. The implication $\widetilde{\text{Until}}_{[a,b\rangle]}(\phi_1,\phi_2) \Rightarrow \widetilde{\text{Until}}_{[a,b\rangle)}(\phi_1,\phi_2)$ is straightforward, so let us assume that $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ee}}\big(\text{Until}_{(0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2), a\big)$ and $b(t) \models_{\mathbb{R}} \text{Until}_{[a,b\rangle]}(\text{true}, \neg\phi_1 \wedge \phi_2)$.

First of all, let us realize that $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ee}}\big(\text{Until}_{(0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2), a\big)$ implies $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ee}}(\phi_1, a)$. In fact, otherwise there would be a $y \in (t, t+a)$ such that $b(y) \models_{\mathbb{R}} \neg\phi_1$; but since it is also $b(y) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2)$ we clearly have a contradiction, as the latter requires in particular that $b(y) \models_{\mathbb{R}} \phi_1$.

Next, let us realize that the facts $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ee}}(\phi_1, a)$ and $b(t) \models_{\mathbb{R}} \text{Lasts}_{\text{ee}}\big(\text{Until}_{(0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2), a\big)$ holding together require that $b(t+a) \models_{\mathbb{R}} \text{Until}_{[0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2)$. In fact, ϕ_1 cannot become false before $t + a$, but it *must* become false somewhere after (or at) $t+a$ to satisfy $\text{Until}_{(0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2, a)$. For the same reason, ϕ_1 must hold until it becomes false after or at $t + a$. All this is exactly what is required by $b(t + a) \models_{\mathbb{R}} \text{Until}_{[0,+\infty))}(\phi_1, \neg\phi_1 \wedge \phi_2)$.

Next, let us consider the consequences of this fact: it means that there exists a $u' \in [t + a, +\infty)$ such that $b(u') \models_{\mathbb{R}} \neg\phi_1 \wedge \phi_2$ and for all $v' \in [t + a, u')$ it is $b(v') \models_{\mathbb{R}} \phi_1$.

Let us first distinguish the case $u' \in [t+a, t+b\rangle$. All in all, ϕ_1 holds over the interval (t, u'), and ϕ_2 holds at u'; therefore, we have $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{[a,b\rangle)}(\phi_1,\phi_2)$.

Otherwise, let us consider the case $u' \notin [t + a, t + b\rangle$; therefore $u' \in \langle' t + b, +\infty)$, where $\langle'$ is the "complement" of $\rangle$. In particular, this implies that for all $v' \in (t, t + b)$ it is $b(v') \models_{\mathbb{R}} \phi_1$. Moreover, we are also assuming that $b(t) \models_{\mathbb{R}} \text{Until}_{[a,b\rangle)}(\text{true}, \neg\phi_1 \wedge \phi_2)$ in this branch of the proof. That is, there exists a $u'' \in [t+a, t+b\rangle$ such that $b(u'') \models_{\mathbb{R}} \neg\phi_1 \wedge \phi_2$. Since $(t, u'') \subseteq (t, t+b)$, then we have shown that $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{[a,b\rangle)}(\phi_1,\phi_2)$, as required. □

When the left bound is zero. Finally, we have to handle the $a = 0$ case for the intervals. Let us start with left-open intervals; the case for left-closed will be an immediate consequence.

Now, we finally need the above results about the expressibility of the $\epsilon\widetilde{\text{NowOn}}$ operator. In fact, we prove the following.

$$\widetilde{\text{Until}}_{(0,b\rangle)}(\phi_1,\phi_2) \;\Leftrightarrow\; \text{Until}_{(0,b\rangle)}(\text{true},\phi_2) \wedge \epsilon\widetilde{\text{NowOn}}\big(\text{Until}_{(0,+\infty))}(\phi_1,\phi_2)\big) \quad (9)$$

Proof of Formula (9). Let us start with the $\Rightarrow$ direction: assume that $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{(0,b\rangle)}(\phi_1,\phi_2)$. That is, there exists a $u \in (t, t + b\rangle$ such that $b(u) \models_{\mathbb{R}} \phi_2$

and, for all $v \in (t,u)$ it is $b(v) \models_{\mathbb{R}} \phi_1$. From $b(u) \models_{\mathbb{R}} \phi_2$ it follows immediately that $\text{Until}_{(0,b\rangle)}(\text{true}, \phi_2)$, so the first conjunct is proved.

Then, let us show that $b(t) \models_{\mathbb{R}} \epsilon\widetilde{\text{NowOn}}\big(\text{Until}_{(0,+\infty))}(\phi_1, \phi_2)\big)$. More precisely, we can show that for all $\alpha \in (t,u)$ it is $b(\alpha) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \phi_2)$. In fact, notice that $u > \alpha$, so $u \in (\alpha, +\infty)$; moreover $[\alpha, u) \subset (t,u)$, as $\alpha > t$. All in all, we have that $b(\alpha) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \phi_2)$.

Let us now consider the $\Leftarrow$ direction, and let us assume that $b(t) \models_{\mathbb{R}} \text{Until}_{(0,b\rangle)}(\text{true}, \phi_2)$. That is, there exists a $u \in (t, t+b\rangle$ such that $b(u) \models_{\mathbb{R}} \phi_2$. If, furthermore, for all $v \in (t,u)$ it is $b(v) \models_{\mathbb{R}} \phi_1$, then the left-hand side holds obviously.

Otherwise, let v' be the smallest instant in (t,u) such that $b(v') \models_{\mathbb{R}} \neg\phi_1$ or $b(v') \models_{\mathbb{R}} \epsilon\widetilde{\text{NowOn}}(\neg\phi_1)$. Recall that we are assuming as hypothesis the right-hand side of the double implication. So, from the definition of the $\epsilon\widetilde{\text{NowOn}}$ operator, we are also assuming that there exists a $\beta > 0$ such that for all $\gamma \in (t, t+\beta)$ it is $b(\gamma) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \phi_2)$. Notice that this implies that for all $\gamma \in (t, t+\beta)$ it is also $b(\gamma) \models_{\mathbb{R}} \phi_1$.

Next, let m be the minimum of the two time distances $\beta/2$ and $(v'-t)/2$. Since $t+m \in (t, t+\beta)$ we have that $b(t+m) \models_{\mathbb{R}} \text{Until}_{(0,+\infty))}(\phi_1, \phi_2)$. That is there exists a $u' \in (t+m, +\infty)$ such that $b(u') \models_{\mathbb{R}} \phi_2$ and for all $v'' \in [t+m, u')$ it is $b(v'') \models_{\mathbb{R}} \phi_1$. But since $m \leq (v'-t)/2$, then we conclude that for all instants t' in (t,u') it is $b(t') \models_{\mathbb{R}} \phi_1$.

Now, if $u' > v'$, we have a contradiction. Thus, it must be $u' \leq v'$. In this case, notice that: $u' \in (t,u) \subset (t, t+b\rangle$, which implies that $b(t) \models_{\mathbb{R}} \widetilde{\text{Until}}_{(0,b\rangle)}(\phi_1, \phi_2)$ as required. □

Then, it is simple to express the case in which the left endpoint is included.

$$\widetilde{\text{Until}}_{[0,b\rangle)}(\phi_1, \phi_2) \;\Leftrightarrow\; \widetilde{\text{Until}}_{(0,b\rangle)}(\phi_1, \phi_2) \vee \phi_2 \tag{10}$$

Finally, notice that Formulas 5 and 6 are valid also when $a = 0 < b$. For $a = b = 0$ it is routine to verify the following, which concludes our set of equivalences.

$$\widetilde{\text{Until}}_{[0,0]\rangle}(\phi_1, \phi_2) \;\Leftrightarrow\; \phi_2 \tag{11}$$

Switching to other semantic models. Let us conclude this section with a remark. Notice that the above equivalence proofs still hold if we make some variations on the semantic model according to which we interpret $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formulas. In particular, the equivalence between strict and non-strict operators holds even for *mono-infinite time domains* (namely, the nonnegative reals $\mathbb{R}_{\geq 0}$), and for the *future-only* fragment of $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO (that is, the fragment which does not use the Since operator). This remark will be useful in comparing the expressiveness of $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO to that of other metric temporal logics.

1.2 $^{\mathbb{R}}_{\mathbb{Z}}$TRIO, MTL and MITL

This section compares $^{\mathbb{R}}_{\mathbb{Z}}$TRIO with the metric temporal logic MTL [Koy90, AH93], and its syntactic fragment MITL [AFH96].

1.2.1 Syntax and Semantics of MTL and MITL

Let us start by introducing formally the syntax and semantics of MTL (and MITL).

Koymans's and Alur and Henzinger's MTL. First of all, let us remark a fact that is usually left implicit (or only briefly hinted at) in the literature. There are actually *two* metric temporal logics that are referred to as "MTL".

The original one is that first defined by Koymans in [Koy90]. Notice that Koymans's MTL permits full quantification on variables, including time variables, as well as the expression of arithmetic relations between time bounds (or other variables). Therefore, as it was originally defined, MTL is a very expressive (and fundamentally undecidable) language which may be compared to full TRIO for several of its features.

Afterward, Alur and Henzinger published several in-depth analyses of the expressiveness of metric temporal logics, including MTL. In particular [AH93] shows that a suitable subset of Koymans's MTL, interpreted over the naturals, can be regarded as an expressively complete (elementarily decidable) fragment of a monadic first-order language over the time sort. Even if [AH93] simply calls "MTL" the proposed logic, it is indeed a proper subset of Koymans's "full" MTL, in that it is purely propositional, and only intervals with constant endpoints are allowed. Much subsequent literature has also used simply the term "MTL" to refer to Alur and Henzinger's MTL *fragment*.

In the same vein, we warn the reader that in this paper the name "MTL" will refer to MTL *à la* Alur and Henzinger, not Koymans's. Moreover, we consider the MTL variant with past operators, as also done in [AH93].

MTL syntax. MTL syntax is defined by the following grammar:

$$\phi \quad ::= \quad \xi \mid \phi_1 \mathsf{U}_I \phi_2 \mid \phi_1 \mathsf{S}_I \phi_2 \mid \neg\phi \mid \phi_1 \wedge \phi_2$$

where I is an interval of $\mathbb{R}_{\geq 0}$[6], whose endpoints are constants.

It is customary to define some derived operators in MTL (as it is done with $^{\mathbb{R}}_{\mathbb{Z}}$TRIO). In Table 1 we list the most common ones. Also notice that we freely introduce abbreviations to denote intervals. Namely, we denote as $\leq u$, $< u$, $\geq l$, $> l$, and $= l$ the intervals $[0, u]$, $[0, u)$, $[l, +\infty)$, $(l, +\infty)$, and $[l, l]$, respectively.

[6] Another difference with respect to most papers dealing with MTL is that they usually consider intervals of the rationals or integers only. These restrictions are however adopted solely to achieve decidability properties, which we do not deal with here; hence, we do not adopt such restrictions — that are however orthogonal to the notion of sampling invariance and to the related issues — in our presentation of MTL.

Operator	Definition	Name
$\Diamond_I \phi$	$\text{true}\mathsf{U}_I \phi$	time-constrained *eventually*
$\Box_I \phi$	$\neg \Diamond_I \neg \phi$	time-constrained *always*
$\phi_1 \mathsf{W}_I \phi_2$	$\neg (\neg \phi_2 \mathsf{U}_I \neg \phi_1)$	time-constrained *unless*
$\overleftarrow{\Diamond}_I \phi$	$\text{true}\mathsf{S}_I \phi$	time-constrained *eventually* (past)
$\overleftarrow{\Box}_I \phi$	$\neg \overleftarrow{\Diamond}_I \neg \phi$	time-constrained *always* (past)
$\phi_1 \mathsf{T}_I \phi_2$	$\neg (\neg \phi_2 \mathsf{S}_I \neg \phi_1)$	time-constrained *unless* (past)

Table 1: MTL derived temporal operators

MTL semantics. In defining MTL semantics, we consider operators that are strict in their first arguments (as it is more common in MTL-related literature), and we adopt bi-infinite temporal domains, in conformance with ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO. Moreover, we consider interpretations over behaviors, adapting the standard interval-based sequence interpretation of MTL (and MITL, see e.g. [AFH96]). Finally, we remark that standard MTL semantics does not use the] variation of the *until* (and *since*) operators, where the first and the second argument are required "to meet", which is instead introduced in ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO.

$b(t) \models_{\mathbb{T}} \xi$	iff	$\xi\vert_{b(t)}$
$b(t) \models_{\mathbb{T}} \phi_1 \mathsf{U}_I \phi_2$	iff	there exists $d \in I$ such that $b(t+d) \models_{\mathbb{T}} \phi_2$ and, for all $u \in (0,d)$ it is $b(t+u) \models_{\mathbb{T}} \phi_1$
$b(t) \models_{\mathbb{T}} \phi_1 \mathsf{S}_I \phi_2$	iff	there exists $d \in I$ such that $b(t-d) \models_{\mathbb{T}} \phi_2$ and, for all $u \in (0,d)$ it is $b(t-u) \models_{\mathbb{T}} \phi_1$
$b(t) \models_{\mathbb{T}} \neg \phi$	iff	$b(t) \not\models_{\mathbb{T}} \phi$
$b(t) \models_{\mathbb{T}} \phi_1 \wedge \phi_2$	iff	$b(t) \models_{\mathbb{T}} \phi_1$ and $b(t) \models_{\mathbb{T}} \phi_2$
$b \models_{\mathbb{T}} \phi$	iff	for all $t \in \mathbb{T}$: $b(t) \models_{\mathbb{T}} \phi$

MITL. MITL [AFH96] is simply defined as a syntactic subset of MTL, where all intervals I in formulas are required to be *non-singular*, i.e., not in the form $[l,l]$ for any $l \in \mathbb{R}_{\geq 0}$.

1.2.2 Equivalence between MTL and ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO

In order to show that MTL and ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO are equally expressive languages, we provide two translations — one from MTL formulas to ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO and the other from ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO formulas to MTL — that preserve truth of formulas. Notice that these translations will make use of the equivalence between strict and non-strict operators shown in Section 1.1; therefore, we will use strict versions of ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO operators whenever needed. For brevity, we omit the proofs that the translations preserve truth of formulas, as they are straightforward by case discussion.

Translating MTL to $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO. The function $\sharp\{\cdot\}$ provides a translation from MTL formulas to $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO ones that preserves truth values.

$$
\begin{array}{lcl}
\sharp\{\xi\} & \equiv & \xi \\
\sharp\{\neg\phi\} & \equiv & \neg\, \sharp\{\phi\} \\
\sharp\{\phi_1 \wedge \phi_2\} & \equiv & \sharp\{\phi_1\} \wedge \sharp\{\phi_2\} \\
\sharp\{\phi_1 \mathsf{U}_I \phi_2\} & \equiv & \widetilde{\mathrm{Until}}_{I\rangle}(\sharp\{\phi_1\}, \sharp\{\phi_2\}) \\
\sharp\{\phi_1 \mathsf{S}_I \phi_2\} & \equiv & \widetilde{\mathrm{Since}}_{I\rangle}(\sharp\{\phi_1\}, \sharp\{\phi_2\})
\end{array}
$$

Translating $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO to MTL. The function $\flat\{\cdot\}$ provides a translation from $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formulas to MTL ones that preserves truth values. In this case, we have to take care of $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO's non-strict operators (which are simply expressible using MTL's strict ones), and to deal with the variation of $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO's operators with a] subscript (which is also rather simply reducible to the standard variation).

$$
\begin{array}{lcl}
\flat\{\xi\} & \equiv & \xi \\
\flat\{\neg\phi\} & \equiv & \neg\, \flat\{\phi\} \\
\flat\{\phi_1 \wedge \phi_2\} & \equiv & \flat\{\phi_1\} \wedge \flat\{\phi_2\} \\
\flat\big\{\mathrm{Until}_{I\rangle}(\phi_1, \phi_2)\big\} & \equiv & \begin{cases}
\flat\{\phi_1\} \wedge (\flat\{\phi_1\}\, \mathsf{U}_I \flat\{\phi_2\}) & \text{if } 0 \notin I \text{ and } \rangle \text{ is }) \\
\flat\{\phi_2\} \vee \big(\flat\{\phi_1\} \wedge \big(\flat\{\phi_1\}\, \mathsf{U}_{(0,u\rangle} \flat\{\phi_2\}\big)\big) & \text{if } 0 \in I = [0,u\rangle \text{ and } \rangle \text{ is }) \\
\flat\{\phi_1\} \wedge (\flat\{\phi_1\}\, \mathsf{U}_I\, \flat\{\phi_2 \wedge \phi_1\}) & \text{if } 0 \notin I \text{ and } \rangle \text{ is }] \\
\flat\{\phi_1 \wedge \phi_2\} \vee \big(\flat\{\phi_1\} \wedge \big(\flat\{\phi_1\}\, \mathsf{U}_{(0,u\rangle}\, \flat\{\phi_2 \wedge \phi_1\}\big)\big) & \text{if } 0 \in I = [0,u\rangle \text{ and } \rangle \text{ is }]
\end{cases} \\
\flat\big\{\mathrm{Since}_{I\rangle}(\phi_1, \phi_2)\big\} & \equiv & \begin{cases}
\flat\{\phi_1\} \wedge (\flat\{\phi_1\}\, \mathsf{S}_I \flat\{\phi_2\}) & \text{if } 0 \notin I \text{ and } \rangle \text{ is }) \\
\flat\{\phi_2\} \vee \big(\flat\{\phi_1\} \wedge \big(\flat\{\phi_1\}\, \mathsf{S}_{(0,u\rangle} \flat\{\phi_2\}\big)\big) & \text{if } 0 \in I = [0,u\rangle \text{ and } \rangle \text{ is }) \\
\flat\{\phi_1\} \wedge (\flat\{\phi_1\}\, \mathsf{S}_I\, \flat\{\phi_2 \wedge \phi_1\}) & \text{if } 0 \notin I \text{ and } \rangle \text{ is }] \\
\flat\{\phi_1 \wedge \phi_2\} \vee \big(\flat\{\phi_1\} \wedge \big(\flat\{\phi_1\}\, \mathsf{S}_{(0,u\rangle}\, \flat\{\phi_2 \wedge \phi_1\}\big)\big) & \text{if } 0 \in I = [0,u\rangle \text{ and } \rangle \text{ is }]
\end{cases}
\end{array}
$$

1.2.3 $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO and MITL

As we recalled above, MITL is the syntactic subset of MTL where intervals are all non-singular. We have shown that MTL is as expressive as $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO. Moreover, as we noticed in Section 1.1, the expression of strict operators using non-strict ones does not require to change the non-singularity of the intervals that are involved. In other words, any MTL formula involving non-singular intervals can be expressed as an $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formula also involving non-singular intervals only. The translation $\flat\{\cdot\}$ shows that the converse also holds: any $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formula involving non-singular intervals only can be rendered as a MTL formula without singular intervals.

All in all, the syntactic restriction on $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formulas that requires that all intervals are non-singular yields a language whose expressiveness coincide with that of MITL's. Therefore, all general results about MITL decidability and expressiveness are retained when dealing with $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formulas without singular intervals — of course, provided the same semantic assumptions (e.g., point-based vs. interval-based, mono-infinite vs. bi-infinite, etc.) are made).

1.3 Expressiveness and Decidability Issues

Now, thanks to the equivalence between $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO and MTL (or MITL) we are able to collocate $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO's relative expressiveness, by drawing from the several works about the expressiveness of MTL and variants thereof that are available in the literature. To this end, Section 6 reviews several relevant works about such an important topic.

Notice, however, that most — if not all — works in the literature consider a semantics for MTL which is different than the one we introduced above. Nonetheless, according to the remark outlined at the end of Section 1.1, our equivalence results still hold for two common semantics, and namely:

- the use of a mono-infinite time domain, namely the nonnegative reals $\mathbb{R}_{\geq 0}$;
- the use of future-only operators (i.e., only Until and not Since as temporal operator).

Therefore, we are able to collocate adequately $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO among other logics interpreted in the interval-based semantics.

An issue that we do not deal with here, for brevity, is how our results can be adapted when dealing with another popular semantic model, namely the point-based semantics (a.k.a. the timed trace). This is an interesting direction which belongs to future work. We notice, however, that the point-based semantics carries several peculiar features that sometimes may make it somewhat unwieldy to model naturally some features of real-time systems (see [HR04] and its summary in Section 6). We believe that this defends our choice of focusing on the interval-based semantics *first*.

Let us conclude this section by adding a couple of notes about the expressiveness of $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO with respect to that of full TRIO, and that of nesting-free $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formulas. Notice that we deliberately stay on an informal level: rather than proving the stated expressiveness results, we just give some evidence, and refer to other works for further details. In this case, we prefer this approach for the sake of brevity and clarity, as a detailed and completely formal analysis of these issues would likely distract us from our main focus.

$\frac{\mathbb{R}}{\mathbb{Z}}$TRIO and TRIO. It is rather obvious that $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO is strictly less expressive than full TRIO. The latter is a very expressive language, which even includes all arithmetic and full first-order quantification. $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO is instead purely propositional, and considers only constant bounds for time intervals. This is enough to separate the two expressive powers.

Nesting operators and expressiveness. An issue which has not been investigated often for metric temporal logics over dense time is how the nesting depth of temporal operators impacts the expressiveness of the resulting language. It is however to be expected that the nesting depth of temporal operators defines an expressively strict hierarchy of formulas, that is each level of nesting introduces a larger class of expressible properties. In particular, it should be possible to prove

that nesting-free $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formulas define a class of properties which is strictly contained in that defined by $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formulas that nest temporal operators at any depth.

Moreover, notice that the results about the expressively strict nesting hierarchy also hold in the discrete time setting. Therefore, it is likely that the same holds for continuous-time behaviors which are restricted by the constraint χ, i.e., of bounded variability, since they can be basically described as discrete histories.

We refer the reader to Section 6 where we summarize some related works about the problem of the expressiveness of nesting with metric temporal logics over dense (and discrete) time. In particular, we mention here the works [AH92a, KS05] for their results, and [BCM05, PD06] for the proof techniques they use, which rely on characterizing the expressiveness of a temporal logic formula given the values of its interval bounds and the nesting depth of its operators.

2 Berkeley and Non-Berkeley Behaviors

This section considers behaviors subject to the regularity constraint χ — introduced in [FR05] — shows some operators that preserve the regularity (Section 2.1), and studies how such a regularity can be characterized when dealing with items varying over dense domains (Section 2.2).

Let us recall that Abadi and Lamport introduced the term *Zeno* [AL94] to identify those behaviors where time converges to a finite value, and thus, in a sense, "stops". The name is after the ancient Greek philosopher Zeno of Elea[7] and his paradoxes on time advancement. In the same vein, we propose to call "*Berkeley* behaviors" those behaviors failing to satisfy a regularity constraint such as χ, from the name of the famous Irish philosopher George Berkeley[8] who criticized the use of the notion of *infinitesimal* among the founding principles of calculus. In fact, in Berkeley behaviors the distance between any two transitions is infinitesimal, that is indefinitely small (or with infimum equal to zero), even if the infinitesimal times may not accumulate (otherwise, the behavior is also Zeno). Therefore, in the remainder of this report we will refer to behaviors satisfying the constraint χ as *non-Berkeley behaviors*.

2.1 Shiftable Operators

The results about the sampling invariance of $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO we presented in [FR05] consider only nesting-free formulas. In general, as we have argued in Section 1.3, this restricts the expressiveness of our formal language. In [FR05] we have also shown how to put any $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formula into a nesting-free one *by introducing auxiliary items*. This does not contradict the results on the expressiveness: in fact the auxiliary items are then subject to the constraint χ, and therefore we end up with a nesting-free formula which is, in general, stronger than the

[7] Circa 490–430 B.C.
[8] 1685–1753 A.D.

original one, as any subformula of the original formula is required to hold over intervals at least as long as δ.

In this respect, we now investigate what kind of $^{\mathbb{R}}_{\mathbb{Z}}$TRIO formulas preserve the χ constraint, or, in other words, what formulas can be nested without need for introducing additional constraints. The basic idea is the following: if, for any behavior b, a formula holding at some instant t is shown to be true over a full δ-length interval that contains t, then the non-Berkeley variability of the basic items can be "lifted" up to the truth value of the formula itself over time, thus allowing one to nest the formula without introducing additional constraints.

Under this respect, we introduce the following definitions, parametric with respect to a parameter $\epsilon > 0$.

Definition 2.1 (ϵ-Shiftable Operator). An n-argument operator Op interpreted over the basic items $\xi_1, \ldots, \xi_n$ is:

- *positively ϵ-shiftable* iff, for all behaviors $b \in [\![\chi_\circ]\!]_{\mathbb{R}}$, whenever $b(t) \models_{\mathbb{R}} \mathrm{Op}(\xi_1, \ldots, \xi_n)$ for some t, there exists an interval $I = [u, u + \epsilon]$ such that $t \in I$ and, for all $v \in I$ it is $b(v) \models_{\mathbb{R}} \mathrm{Op}(\xi_1, \ldots, \xi_n)$;
- *negatively ϵ-shiftable* iff, for all behaviors $b \in [\![\chi_\circ]\!]_{\mathbb{R}}$, whenever $b(t) \models_{\mathbb{R}} \neg\mathrm{Op}(\xi_1, \ldots, \xi_n)$ for some t, there exists an interval $I = [u, u+\epsilon]$ such that $t \in I$ and, for all $v \in I$ it is $b(v) \models_{\mathbb{R}} \neg\mathrm{Op}(\xi_1, \ldots, \xi_n)$;
- *ϵ-shiftable* iff it is both positively and negatively ϵ-shiftable, and its "change points" for some behavior b, that is the instants at which it switches its truth value with respect to b, coincide with some change points of b.[9] In other words, the change points of the operator are a subset (possibly equal to) of the changing points of b.

Therefore, if an operator is δ-shiftable, then its truth value over time respects the constraint χ if its arguments do. Therefore, it can be nested without impacting sampling invariance.

2.1.1 Non-Shiftable Operators

Let us start by showing that — unsurprisingly — not all operators are δ-shiftable. In particular, let us show that the Until operator is not, at least in its most general application. Let ξ_1, ξ_2 be two basic Boolean time-dependent items. Let us consider the behavior b in Figure 1, such that ξ_1 and ξ_2 both hold over $[t, t + \delta]$, and are both false everywhere else. Clearly $b \in [\![\chi_\circ]\!]_{\mathbb{R}}$, and $b(t) \models_{\mathbb{R}} \mathrm{Until}_{[\delta,+\infty)]}(\xi_1, \xi_2)$, but $\mathrm{Until}_{[\delta,+\infty)]}(\xi_1, \xi_2)$ is false throughout $I' = (-\infty, t)$ because ξ_1 is false in any right-neighborhood of a point in I', and it is false through $I'' = (t, +\infty)$, because ξ_2 is always false in the interval $(t + \delta, +\infty)$. Therefore, the Until operator is in general non shiftable.

[9] Notice that this additional requirement on the "change points" is not redundant. For instance the "rigid shift" of a basic item ξ (i.e., $\mathrm{Futr}(\xi, \tau)$) does change its value at instants other than those of the item itself, even if it is both positively and negatively δ-shiftable. We also note that this requirement is similar to that of *stability* introduced elsewhere [Rab03].

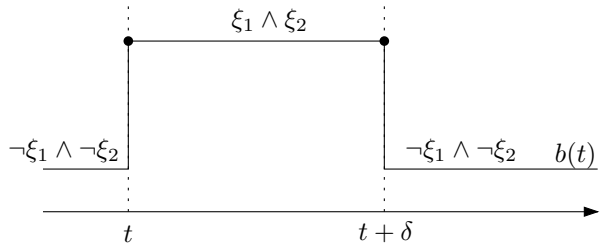

Figure 1: Until is non-shiftable.

In the remainder of this section, we consider some much more restricted applications of the Until (and Since) operator, in particular by avoiding metric constraints and thus expressing only qualitative properties; we are able to show that such restrictions are δ-shiftable.

2.1.2 Qualitative Formulas

Although it is simple to see that "existential" operators — such as WithinF — are positively δ-shiftable under simple restrictions of the bound (e.g., for WithinF(ξ, u) such that it is $u > \delta$), and dually "universal" operators — such as Lasts — are negatively δ-shiftable under the same restrictions, the two facts are in contrast, so that existential operators that are positively δ-shiftable are not negatively δ-shiftable, and vice versa for the universal operators. More explicitly, it is relatively easy to check that WithinF$(\xi, 2\epsilon)$ is positively ϵ-shiftable but not negatively so, and its negation Lasts$(\neg\xi, 2\epsilon)$ is negatively ϵ-shiftable (and not positively so). Therefore, positive and negative shiftability are orthogonal notions, as one does not imply the other.

These considerations suggest that it is the use of *metric* itself that hinders the shiftability of temporal operators. Therefore, we try to consider operators that define *qualitative* properties, and we show that these are indeed fully shiftable.

Definition 2.2 (Qualitative Formulas). A formula ϕ is qualitative iff one of the following applies:

- $\phi = \xi$ for some basic item ξ;
- $\phi = \text{Until}_{I\rangle}(\phi_1, \phi_2)$ with $I = [0, +\infty)$ and ϕ_1, ϕ_2 qualitative formulas;
- $\phi = \text{Since}_{I\rangle}(\phi_1, \phi_2)$ with $I = [0, +\infty)$ and ϕ_1, ϕ_2 qualitative formulas;
- ϕ is a Boolean combination of qualitative formulas

Establishing that qualitative formulas are δ-shiftable amounts to tediously considering several cases: we sketch the proof for the most important ones.

Qualitative Until **is δ-shiftable.** Let ξ_1, ξ_2 be two basic Boolean time-dependent items, and let us show that the formula $\phi = \text{Until}_{[0,+\infty)\rangle}(\xi_1, \xi_2)$ is δ-shiftable.

Positively shiftable. Let $b \in [\![\chi_\circ]\!]_{\mathbb{R}}$ be any non-Berkeley behavior, and let t be an instant such that $b(t) \models_{\mathbb{R}} \phi$. Thus, there exists a $d \in [t, +\infty)$ such that $b(d) \models_{\mathbb{R}} \xi_2$, and for all $u \in [t, d\rangle$ it is $b(u) \models_{\mathbb{R}} \xi_1$. Let us consider several cases:

- if $d \geq t + \delta$, ϕ can be shifted forward over the interval $[t, t + \delta]$;
- if $d < t + \delta$, then there exist t', d' such that $t' \leq t$, $d' \geq d$, $d' - t' = \delta$ and for all $v' \in [t', d']$ it is $b(v') \models_{\mathbb{R}} \xi_1$. Therefore ϕ can be surely shifted over the interval $[t', d)$. Let us further distinguish two cases:
  - if $t - t' \geq \delta$ we are done, as ϕ is positively shiftable over $[t', t]$;
  - otherwise, it must be $d' > d$ and $t - t' < \delta$; therefore notice that ξ_2 cannot switch its value to false before or at d', otherwise there would be two switches within δ time units (against the hypothesis $b \in [\![\chi_\circ]\!]_{\mathbb{R}}$). Hence, ϕ can be shifted over the interval $[t', d']$, and we are done.

This proves that qualitative Until is positively δ-shiftable. For brevity, we omitted some finer-grain details in the above proof sketch, but they can be easily reconstructed by the observant reader.

Negatively shiftable. Let us now show that Until is negatively δ-shiftable. This is the same as proving that $\phi = \text{Releases}_{[0,+\infty)\rangle}(\xi_1, \xi_2)$ is positively δ-shiftable. Note that we drop the negations over ξ_1, ξ_2 as they are inessential in the proof (i.e., the proof has a symmetry with respect to complementing the truth value of primitive conditions). Thus, let $b \in [\![\chi_\circ]\!]_{\mathbb{R}}$ be any non-Berkeley behavior, and let t be an instant such that $b(t) \models_{\mathbb{R}} \phi$: for all $d \in [t, +\infty)$ either $b(d) \models_{\mathbb{R}} \xi_2$ or there exists a $u \in [t, d\rangle$ such that $b(u) \models_{\mathbb{R}} \xi_1$.

Proceeding by contradiction simplifies a bit the proof. Thus let us assume that ϕ is not positively δ-shiftable. In particular, this is the case if there exists $r \in [t, t+\delta]$ such that $\text{Until}_{[0,+\infty)\rangle}(\neg\xi_1, \neg\xi_2)$ holds at r. Therefore, there exists a $d_r \in [r, +\infty)$ such that $b(d_r) \models_{\mathbb{R}} \neg\xi_2$ and for all $u_r \in [r, d_r\rangle$ it is $b(u_r) \models_{\mathbb{R}} \neg\xi_1$. Then, from the definition of the Releases operator, there must be a $u \in [t, d_r\rangle$ such that $b(u) \models_{\mathbb{R}} \xi_1$. To avoid contradictions, it must be $u \in [t, r)$. Moreover, let u' be the largest instant in $[t, r]$ such that $\epsilon\widetilde{\text{UpToNow}}(\xi_1)$ holds at u' (it may be that $u' = u$). Then, in compliance with the regularity constraint χ, for all $v' \in [u' - \delta, u')$ it must be $b(v') \models_{\mathbb{R}} \xi_1$. Notice that $t \in [u' - \delta, u')$. But then, ϕ is shiftable over $[u' - \delta, u')$, and a little reasoning (still based on the properties implied by χ) lets us conclude that we can even extend the interval to a right-closed one (either by including u' or by shifting its left endpoint to the left a bit before $u' - \delta$), so that ϕ is indeed δ-shiftable, against the assumption.

Shiftable. We need a key observation to conclude that Until is δ-shiftable; it is not unlike some considerations we did when proving closure under inverse sampling [FR05]. A δ-shiftable quantitative Until or Since has indeed

the additional property of changing its truth value in correspondence with some basic item changing its value. For instance, it is easy to see that $\phi = \text{Until}_{[0,+\infty)\rangle}(\xi_1, \xi_2)$ changes its value to false exactly when either ξ_2 becomes false, or ξ_1 does. To this end, consider an instant at which $b(t) \models_{\mathbb{R}} \phi$; then there exists a $u \in [t, +\infty)$ such that $b(u) \models_{\mathbb{R}} \xi_2$ and for all $v \in [t, u\rangle$ it is $b(v) \models_{\mathbb{R}} \xi_1$. If we can shift ϕ to the left, then ξ_1 must be true also on some interval of the form $\langle t', t)$ for some $t' < t$; ϕ can only become false when ξ_1 also does, and with the same "edge" (i.e., right-continuously or not). On the other hand, if $u > t$ we can surely shift ϕ to the right, at least until instant u; afterward, ϕ switches to false only if ξ_2 does so, or ξ_1 does, or both. All in all, qualitative Until (and Since) is δ-shiftable. Therefore, the application of a qualitative temporal operator yields a formula whose truth value over time respects the constraint χ *together with the other basic items.*

Boolean combinations are shiftable. That δ-shiftable formulas are closed under complement is apparent from Definition 2.1. Thus, let us just prove that conjunction of formulas preserves δ-shiftability. For a formula $\phi = \phi_1 \wedge \phi_2$ such that ϕ_1 and ϕ_2 are both δ-shiftable, not only do ϕ_1 and ϕ_2 each hold over a δ-length interval, but they also hold over a *common* δ-length interval. This is due to the fact that they both shift values together with some basic items, and these have "change points" that are at least δ time units apart, due to χ. In particular, notice that situations such as the one in Figure 2 cannot happen as the value of b changes from (T, F) to (F, F) and then to (F, T) in less than δ time units. This means that any conjunction ϕ also holds over a δ-length interval. Moreover, note that the same argument about the instants where the truth value change can be lifted the the conjunction formula ϕ itself, as any of ϕ_1 or ϕ_2 becoming false implies that ϕ becomes false there as well. All in all, conjunction is δ-shiftable.

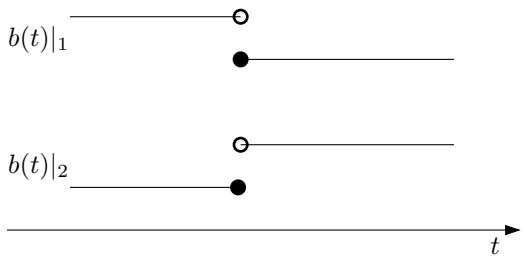

Figure 2: A bi-dimensional behavior not complying with χ.

Left-open intervals are not δ-shiftable. Let us also remark that the adoption of a left-closed interval in the Until is necessary to have δ-shiftable formulas. In fact, Figure 1 serves as a counterexample: $\text{Until}_{(0,\infty)\rangle}(\xi_1, \xi_2)$ only holds in the right-open interval $[t, t + \delta)$, while it is false precisely at $t + \delta$, since ξ_2 is false everywhere in the interval $(t + \delta, +\infty)$.

2.1.3 Sufficiency and (Non) Necessity of Shiftability

Let us now provide the straightforward proofs that having a formula that nests shiftable operators *only* is a sufficient but not necessary condition for the formula to be closed under sampling.

Shiftability is sufficient for closure under sampling. This comes straightforwardly from the definition of δ-shiftability. In fact, let us consider a formula ϕ that nests some operators. Let ϕ' be the formula obtained by putting ϕ in normal form (according to [FR05]), and in particular by "unnesting" all operators by introducing additional items $\xi'_1, \xi'_2, \ldots$.

Then, let us consider any of these additional items ξ'_i. In the normal form, we have a constraint of the form $\xi'_i \Leftrightarrow \mathrm{Op}(\cdots)$, where Op is a δ-shiftable operator by hypothesis. Therefore, the constraint χ simply evaluates to true for the item ξ'_i. In other words, the item ξ'_i satisfies all requirements for closure under sampling; in particular, the formulas in which it is mentioned are closed under sampling. Since the same applies to any additional item ξ'_i, all in all the original formula ϕ is closed under sampling, with the constraint χ applying to basic items only.

Shiftability is not necessary for closure under sampling. Let us consider the formula $\zeta = \mathrm{Lasts}_{\mathrm{ee}}(\xi, \tau)$, for any $\tau > \delta$. Clearly, ζ is not δ-shiftable according to the definition, as the item ξ is not constrained to stay true out of the τ-length interval introduced by the $\mathrm{Lasts}_{\mathrm{ee}}$ operator. Nonetheless, it is easy to see that $b \models_{\mathbb{R}} \zeta$ is the same as $b \models_{\mathbb{R}} \mathrm{Alw}(\xi)$. The formula $\mathrm{Alw}(\xi)$ is obviously sampling invariant, and so we are done.[10]

Shiftability and closure under inverse sampling. Notice that δ-shiftability is a requirement on formulas interpreted over continuous time, whereas the notion of closure under inverse sampling applies to formulas interpreted over discrete time. Nonetheless, the above results about shiftability have consequences also to closure under inverse sampling. In fact, it is easy to see that whenever a formula ϕ nests only subformulas $\varphi_1, \varphi_2, \ldots$ all of whose adaptations $\eta_\delta^{\mathbb{Z}}\{\varphi_1\}, \eta_\delta^{\mathbb{Z}}\{\varphi_2\}, \ldots$ are δ-shiftable, then ϕ is closed under inverse sampling. Thus, shiftability of the discrete-to-continuous adaptation of operators is a sufficient condition for a formula to be closed under inverse sampling. It is also straightforward to realize that the condition is not necessary (we could provide examples along the same lines as the one above about closure under sampling).

Finally, let us point out that this requirement cannot be moved to the non-adapted formula itself, for qualitative formulas. In other words, there are qualitative formulas whose discrete-to-continuous adaptations are not δ-shiftable (thus, in particular, are not qualitative formulas). Therefore, in general nesting qualitative formulas in discrete-time formulas cannot be done "for free" — that

[10] As an aside, let us point out that the definition of sampling invariance [FR05, FR06] implies that whenever two formulas ϕ_1, ϕ_2 are equivalent both in dense and in discrete time, and so are their adaptations (i.e., $\eta_\delta^{\mathbb{R}}\{\phi_1\} \equiv \eta_\delta^{\mathbb{R}}\{\phi_2\}$ in discrete time, and $\eta_\delta^{\mathbb{Z}}\{\phi_1\} \equiv \eta_\delta^{\mathbb{Z}}\{\phi_2\}$ in dense time), then ϕ_1 is sampling invariant if and only if ϕ_2 is also sampling invariant.

is without introducing additional constraints on the variability of the adapted formula — for the sake of closure under inverse sampling. As a proof of this fact, let us consider the formula $\phi = \mathrm{Until}_{[0,+\infty)]}(\xi_1, \xi_2)$ and its discrete-to-continuous adaptation $\phi' = \eta_\delta^{\mathbb{Z}}\{\mathrm{Until}_{[0,+\infty)]}(\xi_1, \xi_2)\}$. It is easy to see that ϕ' is equivalent to $\mathrm{WithinP}_{\mathrm{ie}}(\xi_2, \delta) \vee \mathrm{Until}_{[0,+\infty)]}(\xi_1, \xi_2)$; let us show that ϕ' is not δ-shiftable. To this end, let us consider the behavior b represented by the interval-based sequence $\langle(-\infty, 0.7], \xi_2 \wedge \neg\xi_1\rangle, \langle(0.7, +\infty), \neg\xi_2 \wedge \xi_1\rangle$. Clearly, ϕ' is positively and negatively δ-shiftable over b, since it is true in the interval $(-\infty, \delta + 0.7]$ and false in the complement interval $(\delta + 0.7, +\infty)$. However, its truth value changes at $\delta + 0.7$, which is not a change point for either ξ_1 or ξ_2 (as $\delta > 0$). Therefore, ϕ' is not (fully) δ-shiftable.

2.1.4 A Formula Not Closed Under Sampling

Let us now exhibit a formula ϕ^{ns} which nests non-shiftable operators and, in fact, is not closed under sampling.

$$\phi^{\mathrm{ns}} \quad \equiv \quad \mathrm{Som}(\mathrm{Lasts}_{\mathrm{ii}}(\xi, \delta))$$

Let us consider the behavior $b \in [\![\chi_\circ]\!]_{\mathbb{R}}$ of Figure 3 — where we assume that ξ is false everywhere except that in some interval, larger than δ and internal to $(t, t + 2\delta)$ — as a proof (by counterexample). Remember that the definition of sampling invariance requires invariance for any choice of the origin z; hence, let us choose adversarially the sampling as in the figure. Clearly $b \models_{\mathbb{R}} \phi^{\mathrm{ns}}$;

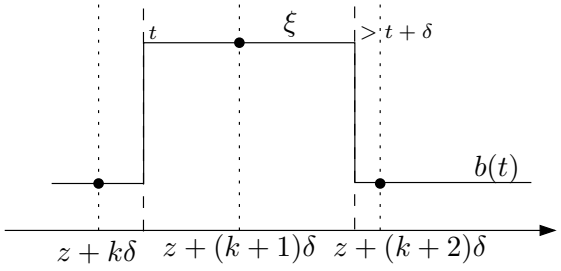

Figure 3: Proof of non sampling invariance.

nonetheless, the adaptation $\eta_\delta^{\mathbb{R}}\{\phi^{\mathrm{ns}}\}$ of ϕ^{ns} is:

$$\eta_\delta^{\mathbb{R}}\{\phi^{\mathrm{ns}}\} \quad = \quad \mathrm{Som}(\mathrm{Lasts}_{\mathrm{ii}}(\xi, 1))$$

It is easy to realize that $\mathrm{Lasts}_{\mathrm{ii}}(\xi, 1) = \xi_1 \wedge \mathrm{NowOn}(\xi_1)$ holds at no discrete sampling point of $\sigma_{\delta,z}[b]$, so $\sigma_{\delta,z}[b] \not\models_{\mathbb{Z}} \eta_\delta^{\mathbb{R}}\{\phi^{\mathrm{ns}}\}$. That is, ϕ^{ns} is not closed under sampling, and *a fortiori* nor sampling invariant.

With a very similar proof, it can be seen that the regularity constraint χ itself is not a sampling-invariant formula (in particular, it is not closed under sampling).

2.2 Towards Characterizing Berkeley Behaviors of Dense-Valued Items

This section aims at characterizing non-Berkeley behaviors for basic items that map to a dense (or, more specifically, continuous) domain. More precisely, we aim at giving a mathematical characterization of the behaviors that satisfy the constraint $\chi_{\bullet}^{\phi}$ for dense-valued items.

As we have already discussed in [FR05, FR06], the behavior constraint $\chi_{\bullet}^{\phi}$ for dense-valued items depends, in general, on the particular formula ϕ that constitutes our specification; the notation $\chi_{\bullet}^{\phi}$ stresses this fact. In general, the specification formula ϕ can be very complex, and so can be the conditions in Ξ. Therefore, the actual "physical" impact of the constraint $\chi_{\bullet}^{\phi}$ may be very difficult to predict and characterize in a clear-cut manner.

To focus our discussion, let us choose a particular — yet significant and rather general — form for the conditions in Ξ. Namely, we consider conditions of the form $\mathsf{x} \in \langle \mathrm{l}, \mathrm{u} \rangle$, where x is a basic item (for simplicity, taking values to the set $\mathbb{R}$), $\mathrm{l}, \mathrm{u} \in \mathbb{R} : \mathrm{l} < \mathrm{u}$ are two constant bound values, and $\langle \in \{(, [\}$ (resp. $\rangle \in \{),]\}$) denotes whether the left (resp. right) endpoint in excluded or included. We point out that the generalization to $n > 1$ items is straightforward. In this basic case, the condition $\chi_{\bullet}^{\phi}$ becomes:

$$\chi_{\bullet}^{\phi} \;=\; \mathrm{WithinP}_{\mathrm{ii}}(\mathrm{Lasts}_{\mathrm{ii}}(\mathsf{x} \in \langle \mathrm{l}, \mathrm{u} \rangle, \delta), \delta) \vee \mathrm{WithinP}_{\mathrm{ii}}(\mathrm{Lasts}_{\mathrm{ii}}(\mathsf{x} \notin \langle \mathrm{l}, \mathrm{u} \rangle, \delta), \delta)$$

Let us now consider a generic behavior $b \in \mathcal{B}_{\mathbb{R}}$ for x, that is a generic function $b : \mathbb{R} \to \mathbb{R}$. We now give an "approximate" mathematical characterization of the subset $[\![\chi_{\bullet}^{\phi}]\!]_{\mathbb{R}} \subset \mathcal{B}_{\mathbb{R}}$ for any ϕ where Ξ is in the form we have outlined.

Notice that we will not give a completely equivalent characterization since, as we will show, there are some aspects implied by $\chi_{\bullet}^{\phi}$ which are hard to characterize in a simple way. Nonetheless, we are going to provide a mathematical notion that can be reasonably regarded as a formal description of the behaviors satisfying $\chi_{\bullet}^{\phi}$, *within some tolerance*.

2.2.1 Uniformly Continuous Functions

Let us start by defining formally the notion of *uniform continuity*. As we will show, it will constitute our characterization of non-Berkeley behaviors.

Continuity. First, let us recall the well-known definition of continuous function. A function $b : \mathbb{R} \to \mathbb{R}$ is *continuous at some point* t iff: for any $\epsilon > 0$ there exists a $\delta_{\epsilon,t} > 0$ such that for all x such that $|x - t| < \delta_{\epsilon,t}$ it is: $|f(x) - f(t)| < \epsilon$. Notice that in the definition $\delta_{\epsilon,t}$ is in general a function not only of ϵ, but also of t, and the notation stresses this fact.

Then, a function b is continuous *over an interval* $I \subseteq \mathbb{R}$ iff it is continuous at all points in I.

Uniform continuity. A function $b : \mathbb{R} \to \mathbb{R}$ is *uniformly continuous over an interval* $I \subseteq \mathbb{R}$ iff: for any $\epsilon > 0$ there exists a $\delta_\epsilon > 0$ such that, for all $x, t \in I$ such that $|x - t| < \delta_\epsilon$ it is: $|f(x) - f(t)| < \epsilon$. Notice that the key difference with respect to the definition of continuity is that now δ_ϵ does not depend on the chosen t, but it must be unique for all points in I.

Notice that uniform continuity is a notion close to, but more general, than having bounded derivative. In fact, one can show that any function with bounded derivative (i.e., such that $\sup_{x \in I} |b'(x)| < K$ for some K) is uniformly continuous. But the converse is in general not true: for instance, the function $\sqrt{x}$ has derivative $1/(2\sqrt{x})$ which is unbounded over the open interval $(0, 1)$ as it goes to ∞ as x goes to 0. Nonetheless, it is not difficult to see that $\sqrt{x}$ is uniformly continuous over $(0, 1)$.

2.2.2 Sampling a Uniformly Continuous Behavior

Let us now show what is the link between uniformly continuous functions and the slow-variability requirement expressed by $\chi^\phi_\bullet$.

Uniformly continuous behaviors behave. Let us consider a condition Ξ of the form $\mathsf{x} \in \langle \mathrm{l}, \mathrm{u} \rangle = I$; if b is uniformly continuous over $\mathbb{R}$, then it is always possible to find a δ such that $|b(t') - b(t)| < |I| = \mathrm{u} - \mathrm{l}$ for all instants t, t' such that $|t' - t| < \delta$. Therefore, let such δ be our sampling period. Then, let us consider any instant at t which b "enters" the interval I such that b is monotonic non-decreasing over I; then b will remain in I for at least δ time units before crossing it and exiting above. Thus, for monotonic behaviors, the condition $\chi^\phi_\bullet$ is always satisfied.

More generally, however, b is non monotonic. In this case, let us pick a $2\epsilon < \mathrm{u} - \mathrm{l}$ and require that every behavior we consider does not do the following: enter I, reach at least $\mathrm{l} + \epsilon$, exit I again, all this in less than a sampling period. Then, if b is uniformly continuous, we always have a δ_ϵ which depends on 2ϵ, such that, if we pick δ_ϵ as sampling period, b satisfies the requirement.

In general, ϵ can be as small as desired: uniform continuity ensures that a suitable δ_ϵ can always be found. Obviously, this can be generalized to several intervals partitioning the domain of x, by considering the smallest of such intervals in place of the unique I. We assume that the various intervals are derived from a specification formula ϕ which is finite in size; therefore the number of such intervals will always be finite as well (and thus the notion of "smallest" is well-defined).

Undesired behaviors about extrema. Although what discussed above allows one to find behaviors that comply with the regularity constraint $\chi^\phi_\bullet$ to within some tolerance (given by the choice of ϵ), it is also clear that, however small ϵ is, one can always find behaviors that do not comply with $\chi^\phi_\bullet$ but nonetheless are uniformly continuous. Figure 4 gives a graphical representation of this fact: if b enters the interval I "just a little bit", reaches a maximum at

$\hat{t} < \epsilon$, and exits it afterward, it can do so within less than any fixed δ while still having bounded derivative. So, uniform continuity is not precisely a character-

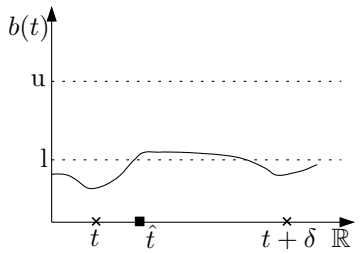

Figure 4: A behavior violating $\chi^{\phi}_{\bullet}$ about an extremum.

ization of the requirement expressed by $\chi^{\phi}_{\bullet}$, but only within some arbitrarily small tolerance, given by the choice of ϵ.

Sets of behaviors. One more thing should be said about the use of the notion of uniform continuity to characterize behaviors satisfying $\chi^{\phi}_{\bullet}$. So far, we have only considered the uniform continuity of a *single* behavior b. However, one would like to derive similar properties about a whole *set of behaviors* $B = \{b_i\}$. In general, even if each $b_i \in B$ is uniformly continuous, the δ_ϵ for b_i may also depend on i, so it is actually a $\delta_{i,\epsilon}$. If, for a given ϵ, $\inf_i \delta_{i,\epsilon}$ is zero, then it is not possible to find a unique sampling period δ such that all samplings of any behavior in B with period δ satisfy $\chi^{\phi}_{\bullet}$ (up to the chosen tolerance ϵ). Therefore, we are going to include explicitly that such an infimum is greater than zero, to rule out undesirable cases.

Quasi-Non-Berkeley behaviors. All in all, according to what we have discussed above and generalizing, we introduce the following terminology. Let m be a basic time-dependent item mapping to the set $D \equiv D_1 \times \cdots \times D_n$, such that every D_i is a continuous set. Let b be a behavior mapping $\mathbb{R}$ to the set D. If b is uniformly continuous in $\mathbb{R}$, then we say that b is *quasi-non-Berkeley*.

A set $B = \{b_i\}$ of behaviors is quasi-non-Berkeley if each $b_i \in B$ is uniformly continuous, and, for all $\epsilon > 0$, the infimum $\inf_i \delta_i(\epsilon)$ of the δ_i's (given by the uniform continuity requirement for each b_i), is strictly greater than zero.

Clearly, a non-Berkeley behavior is *a fortiori* a quasi-non-Berkeley one. However, we have outlined above behaviors that are compatible with being quasi-non-Berkeley but are not fully non-Berkeley (i.e., they fail to satisfy $\chi^{\phi}_{\bullet}$).

Also notice that we are unable to express the quasi-non-Berkeley requirement in $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO as we did for the non-Berkeley requirement. In the former case we were interested in the orthogonal concern of giving a mathematical characterization of the behaviors, rather than one easily expressible in temporal logic.

2.2.3 Continuity, Uniform Continuity, and Analyticity

Gargantini and Morzenti discussed in [GM01] how the non-Zeno requirement for dense-valued items can be mathematically characterized by the notion of analyticity. We know that, for discrete-valued items, the non-Berkeley requirement is strictly stronger than the non-Zeno requirement: while the former requires that the lower bound on the constancy intervals of the items is strictly positive, the latter just requires that intervals of infinitesimal length do not accumulate.

In this vein, let us make a brief mathematical detour to show the relations between the notions of continuity, uniform continuity, and analyticity to understand how non-Zenoness and non-Berkleyness are related for dense-valued items. In particular, we show why uniform continuity and analyticity are orthogonal notions. In the remainder, unless otherwise stated, we consider the whole real axis $\mathbb{R}$ as the domain of functions.

Analytic and not uniformly continuous. The simple polynomial function $f_1(x) = x^2$ is trivially analytic. However, its first-order derivative $f_1'(x) = x$ is not bounded, and indeed f_1 is not uniformly continuous.

Uniformly continuous and not differentiable. It is apparent from the definitions that a uniformly continuous function is also *a fortiori* continuous. However, there exist functions which are uniformly continuous but are not even differentiable (and thus *a fortiori* not even C^1-smooth and thus clearly not analytic). The following function f_2 is an example of such functions.

$$f_2(x) \quad = \quad |x - [x]|$$

where $[\cdot]$ denotes the nearest integer function:

$$[x] \quad = \quad \begin{cases} \lfloor x \rfloor & \text{if } x \leq \lfloor x \rfloor + 1/2 \\ \lceil x \rceil & \text{otherwise} \end{cases}$$

$f_2(x)$ indicates the distance from the integer which is nearest to x; as it can be seen clearly from the plot of the function if Figure 5, f_2 is not differentiable at all integer and half-integer[11] points.

Uniformly continuous, C^∞-smooth, and not analytic. Let us now show an example of function which is uniformly continuous, has continuous derivatives of all orders (i.e., it is C^∞-smooth), but nonetheless fails to be analytic.

$$f_3(x) \quad = \quad \begin{cases} 0 & \text{if } x = 0 \\ e^{-1/x^2} & \text{if } x \neq 0 \end{cases}$$

Figure 6 shows a plot of f_3, for reference.

[11] I.e., points $k + 1/2$ for some $k \in \mathbb{Z}$.

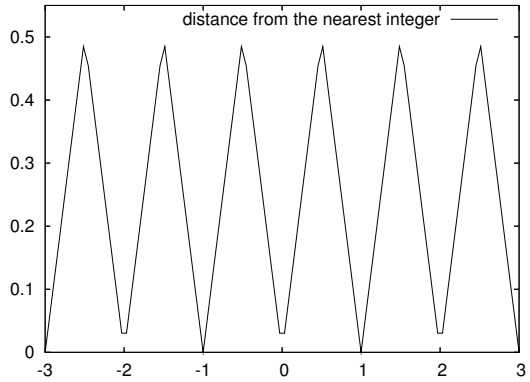

Figure 5: A function f_2 which uniformly continuous but not differentiable.

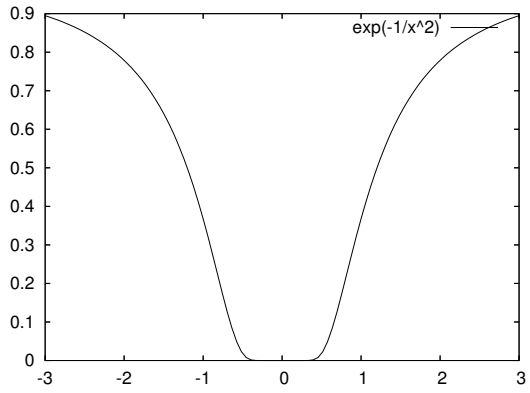

Figure 6: A function f_3 which is uniformly continuous and C^∞, but not analytic.

First of all, it can be shown by induction that the n-th order derivative of f_3 is of the form:

$$f_3^{(n)}(x) \quad = \quad \begin{cases} 0 & \text{if } x = 0 \\ \rho_n(x) e^{-1/x^2} & \text{if } x \neq 0 \end{cases}$$

where $\rho_n(x)$ is some rational function without singularities in $\mathbb{R}_{\neq 0}$; therefore, each derivative is continuous over $\mathbb{R}_{\neq 0}$, and the function is infinitely differentiable. Moreover, the form of the derivative implies that its limit as x goes to zero is also zero, so all the derivatives are indeed continuous over all $\mathbb{R}$.

However, f_3 is not analytic at the origin. In fact, since $f_3^{(n)}(0) = 0$ for all n, the Taylor series of f_3 at 0 is simply the constant 0. Therefore, the series about the origin does not converge to the function.

Finally, let us demonstrate that f_3 is uniformly continuous. To this end, let us just consider the first-order derivative f_3' of f_3:

$$f_3'(x) \quad = \quad \begin{cases} 0 & \text{if } x = 0 \\ \frac{2}{x^3} e^{-1/x^2} & \text{if } x \neq 0 \end{cases}$$

Without a detailed analysis, let us just look at the plot of f_3' in Figure 7 and see that it is indeed bounded; therefore f_3 is uniformly continuous, having bounded derivative.

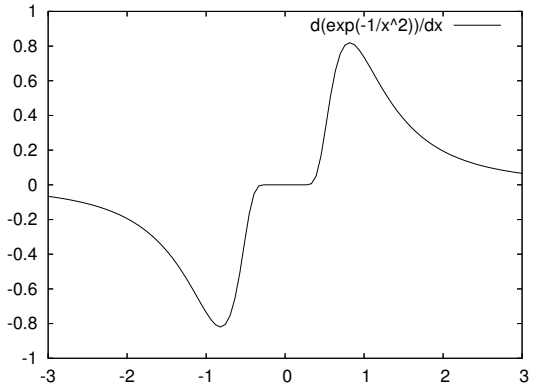

Figure 7: The derivative f_3' of f_3 is bounded everywhere.

Uniform continuity over finite time. Let us also recall a classic result of mathematical analysis which goes under the name of *Heine-Cantor theorem*. The theorem states that every continuous function defined over a compact (metric) domain is also uniformly continuous over the same domain. In particular, over the real line every closed interval is a compact set, thus every continuous function is also uniformly continuous over a closed interval. This means that if

we restrict ourselves to behavior over *finite time*, and we ensure that the chosen finite interval is also closed, then it is sufficient to require that our behaviors are continuous functions to meet the quasi-non-Berkeley requirement.

3 A Comparison With Digitization

Henzinger, Manna and Pnueli [HMP92] were the first ones — to the best of our knowledge — to study explicitly the continuous- and discrete-time semantics of timed systems in general, and temporal logic in particular. This section briefly reports the main results of [HMP92] and relate them to our results about sampling invariance [FR05, FR06].

There are two main differences between our approach and the one in [HMP92].

- First, while the framework in [HMP92] adopts a point-based semantics based on (timed) trace (called *timed state sequence* in [HMP92]), ours refers to total functions from time to the state domain (i.e., behaviors). In practice, since it is customary to restrict to non-Zeno behaviors, we can equivalently say that our framework has interval-based sequences as its semantic structure [GM01, HR04].

- Second, while sampling invariance is defined for a *formula*, digitization is defined for a *property*, that is a set of timed traces. In other words, digitization is defined independently of any language, whereas sampling invariance is defined with respect to the metric temporal logic $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO.

Under this respect, the next Section 3.1 introduces the framework in [HMP92], while Section 3.2 suggests how it can be compared to ours. Afterward, Section 3.3 carries out the comparison under those lines, and shows that the two notions of sampling invariance and digitization are *orthogonal*. Finally, Section 3.4 sketches other aspects that have not been touched upon in the comparison, and may belong to future work.

3.1 Timed Traces and Digitization

Let us consider a generic timed trace; while timed traces are usually defined as mono-infinite sequences, our framework refers to bi-infinite time domains. Therefore, we change the definitions of [HMP92] to refer them to bi-infinite sequences. It is routine to check that all the relevant results are retained in the new setting. Moreover, we could have equivalently rephrased all our framework in terms of mono-infinite sequences, obtaining basically the same results in the comparison.

Timed traces. So, let us consider the generic timed trace:

$$\rho: \quad \cdots(\sigma_{-2},\tau_{-2})(\sigma_{-1},\tau_{-1})(\sigma_0,\tau_0)(\sigma_1,\tau_1)(\sigma_2,\tau_2)\cdots$$

where $\tau_i \in \mathbb{R}$ for all $i \in \mathbb{Z}$. Following [HMP92], we assume *weak monotonicity* of the timestamps (i.e., $\tau_i \leq \tau_{i+1}$ for all $i \in \mathbb{Z}$), and *progress* (i.e., for all $t \in \mathbb{R}$ there exists a $i \in \mathbb{Z}$ such that $\tau_i \geq t$).

ϵ-digitizations. For any $0 \leq \epsilon < 1$, and any timed trace ρ, the *ϵ-digitization* $[\rho]_\epsilon$ is the integer-timed trace that results from rounding, with respect to ϵ, every timestamp in ρ to an integer value. Namely:

$$[\rho]_\epsilon : \quad \cdots(\sigma_{-2}, [\tau_{-2}]_\epsilon)(\sigma_{-1}, [\tau_{-1}]_\epsilon)(\sigma_0, [\tau_0]_\epsilon)(\sigma_1, [\tau_1]_\epsilon)(\sigma_2, [\tau_2]_\epsilon)\cdots$$

where:

$$[x]_\epsilon \quad = \quad \begin{cases} \lfloor x \rfloor & \text{if } x \leq \lfloor x \rfloor + \epsilon \\ \lceil x \rceil & \text{otherwise} \end{cases}$$

Digitization of a property. Given a property π, that is a set of timed traces, its *digitization* $[\pi]$ is a property containing all and only integer-timed traces resulting from digitizing all traces in π with respect to all fractional values $0 \leq \epsilon < 1$:

$$[\pi] \quad = \quad \{[\rho]_\epsilon \mid \rho \in \pi \text{ and } 0 \leq \epsilon < 1\}$$

Digitizable properties. Given a property π, we say that:

- π is *closed under digitization* iff $[\pi] \subseteq \pi$ (or, equivalently, for all timed traces ρ: $\rho \in \pi$ implies $[\rho] \subseteq \pi$);
- π is *closed under inverse digitization* iff, for all timed traces ρ, $[\rho] \subseteq \pi$ implies $\rho \in \pi$;
- π is *digitizable* iff it closed under both digitization and inverse digitization (or, equivalently, for all timed traces ρ: $\rho \in \pi$ iff $[\rho] \subseteq \pi$).

3.2 Digitizable Formulas

Let us now suggest a way to compare the two framework of sampling invariance and digitizability. First of all, let us define an $^{\mathbb{R}}_{\mathbb{Z}}$TRIO semantics over timed traces, that mirrors as closely as possible that over behaviors. Second, let us give a definition of what it means for an $^{\mathbb{R}}_{\mathbb{Z}}$TRIO *formula* to be digitizable.

Timed trace semantics of $^{\mathbb{R}}_{\mathbb{Z}}$TRIO. Let us define the satisfiability relation for $^{\mathbb{R}}_{\mathbb{Z}}$TRIO formulas and a generic timed trace ρ. Let $i \in \mathbb{Z}$ be a generic position in the trace. Then:[12]

[12] Sums between a real value and an interval are meant to be Minkovsky sums, with the familiar meaning.

$\rho_i \models \xi$	iff	$\xi \in \sigma_i$
$\rho_i \models \mathrm{Until}_{I\rangle}(\phi_1, \phi_2)$	iff	there exists $j \geq i$ such that: $\tau_j \in \tau_i + I$, $\rho_j \models \phi_2$, and for all integers $k \in [i, j\rangle$: $\rho_k \models \phi_1$
$\rho_i \models \mathrm{Since}_{I\langle}(\phi_1, \phi_2)$	iff	there exists $j \leq i$ such that: $\tau_j \in \tau_i - I$, $\rho_j \models \phi_2$, and for all integers $k \in \langle j, i]$: $\rho_k \models \phi_1$
$\rho_i \models \neg\phi$	iff	$\rho_i \not\models \phi$
$\rho_i \models \phi_1 \wedge \phi_2$	iff	$\rho_i \models \phi_1$ and $\rho_i \models \phi_2$
$\rho \models \phi$	iff	for all $i \in \mathbb{Z}$: $\rho_i \models \phi$

Digitizable formulas. Given an $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formula ϕ, we say that:

- ϕ is *closed under digitization* iff the property $\{\rho \mid \rho \models \phi\}$ is closed under digitization;
- ϕ is *closed under inverse digitization* iff the property $\{\rho \mid \rho \models \phi\}$ is closed under inverse digitization;
- ϕ is *digitizable* iff the property $\{\rho \mid \rho \models \phi\}$ is digitizable.

3.3 Digitization and Sampling Invariance Are Orthogonal Notions

This section shows that the two notions of digitization and sampling invariance define incomparable classes of formulas, when the comparison is carried out along the lines introduced in the previous sections. To this end, first of all we exhibit a formula which is sampling invariant but not digitizable; then, we provide a formula which is digitizable but not sampling invariant.

We remark that, throughout this section, we always refer to the notion of sampling invariance according to the adaptation functions $\eta_\delta^{\mathbb{R}}\{\}, \eta_\delta^{\mathbb{Z}}\{\}$ as we defined them in [FR05]. Exploring the consequences of adopting different definitions of adaptation functions in the comparison with digitization is out of the scope of the present work.

3.3.1 Sampling Invariant Non-Digitizable Formulas

It is simple to find sampling invariant non-digitizable formulas, as they can be build directly out of some examples of non-digitizable formulas given in [HMP92]. In particular, let us consider the formula:

$$\Theta \quad \equiv \quad \mathrm{Som}(\mathrm{WithinF}_{\mathrm{ie}}(\xi, 1))$$

It is simple to check that Θ is sampling invariant, since the outer existential quantification on time (i.e., the Som operator) does not affect sampling invariance.

On the other hand, Θ is not closed under digitization. The timed trace:

$$\rho' \quad = \quad \cdots(\neg\xi, k)(\xi, k + \mu)(\neg\xi, \tau)\cdots$$

with $k \in \mathbb{Z}$, $0 < \mu < 1$, $\tau > k+1$, and such that ξ is false at any other position in the trace, provides a proof of this (in the form of a counterexample). In fact, ρ' satisfies Θ, but any of its ϵ-digitizations with respect to any $\epsilon < \mu$ does not, as they all correspond to $\cdots(\neg\xi, k)(\xi, k+1)(\neg\xi, \tau')\cdots$ with $\tau' \geq k+1$.

3.3.2 Digitizable Non-Sampling Invariant Formulas

Let us consider the following ${}^{\mathbb{R}}_{\mathbb{Z}}$TRIO formula (also put in normal form).

$$\Upsilon \quad \equiv \quad \mathrm{Som}\big(\xi \wedge \neg \mathrm{Until}_{(0,+\infty))}(\xi, \mathrm{true})\big) \equiv \mathrm{Som}\big(\xi \wedge \mathrm{Releases}_{(0,+\infty))}(\neg\xi, \mathrm{false})\big)$$

Let us also build the the formula:

$$\Omega \quad \equiv \quad \Upsilon \wedge \phi^{\mathrm{ns}}$$

where ϕ^{ns} was defined in Section 2.1.4. We now show that Ω is not sampling invariant, but it is digitizable.

Ω is digitizable. Let us write out explicitly the semantics of Υ, when interpreted over timed traces according to the weak-time semantics defined beforehand. For a timed trace $\rho = (\sigma, \tau)$: $\rho \models \Upsilon$ iff there exists a $i \in \mathbb{Z}$ such that:

1. $\sigma_i \models \xi$, and, for all $j \geq i$:
2. either $\sigma_j \not\models \mathrm{true}$ and $\tau_j > \tau_i$, which however never holds,
3. or $\sigma_k \not\models \xi$, for some $i \leq k < j$.

Now, since condition 3 must hold for any $j \geq i$, in particular (for $j = i+1$) it requires that $\sigma_i \not\models \xi$. This is in contradiction with condition 1, therefore $\Upsilon \equiv$ false in the timed trace semantics.

Thus also $\Omega \equiv \Upsilon \wedge \phi^{\mathrm{ns}} \equiv \mathrm{false} \wedge \phi^{\mathrm{ns}} \equiv \mathrm{false}$. The formula false is trivially closed under digitization, since so is the empty set of timed traces $\emptyset$. Moreover, it is also trivially closed under inverse digitization, therefore Ω is digitizable.

Ω is not sampling invariant. Let us now consider the adaptation of Υ:

$$\eta^{\mathbb{R}}_{\delta}\{\Upsilon\} \quad \equiv \quad \mathrm{Som}\big(\xi \wedge \mathrm{Releases}_{(0,+\infty)]}(\neg\xi, \mathrm{false})\big)$$

It is not difficult to realize that, thanks to the adaptation of the Releases subformula to the] bracket, $\eta^{\mathbb{R}}_{\delta}\{\Upsilon\}$ is satisfiable in discrete time.[13]

Then, let us consider the behavior b in Figure 8, which is derived from the one in Figure 3 by additionally assuming that ξ holds at t and at $t+\mu$ as well. (In the figure, crosses now denote sampling instants). Now, clearly $b(t+\mu) \models_{\mathbb{R}} \xi \wedge \mathrm{Releases}_{(0,+\infty))}(\neg\xi, \mathrm{false})$, and $b(t) \models_{\mathbb{R}} \mathrm{Lasts}_{\mathrm{ii}}(\xi, \delta)$, therefore $b \models_{\mathbb{R}} \Omega \wedge \chi_{\circ}$. However, while $\sigma_{\delta,z}[b] \models_{\mathbb{Z}} \eta^{\mathbb{R}}_{\delta}\{\Upsilon\}$, we have shown in Section 2.1.4 that $\sigma_{\delta,z}[b] \not\models_{\mathbb{Z}} \eta^{\mathbb{R}}_{\delta}\{\phi^{\mathrm{ns}}\}$. Therefore $\sigma_{\delta,z}[b] \not\models_{\mathbb{Z}} \Omega$, and thus Ω is not closed under sampling, and thus *a fortiori* it is not sampling invariant.

[13] Therefore, the particular definition of $\eta^{\mathbb{R}}_{\delta}\{\}$ plays an important role here. As usual, we do not discuss (for the sake of brevity) the impact of choosing different definitions for $\eta^{\mathbb{R}}_{\delta}\{\}$.

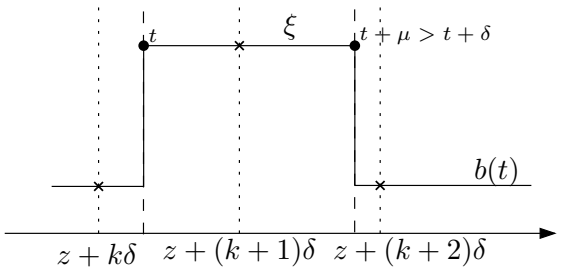

Figure 8: Proof of non sampling invariance.

3.4 Other Aspects for Comparisons

Let us now briefly hint at other aspects that may pertain to a comparison between the notions of sampling invariance and digitizability, but which have not been addressed here. We present them in a rather informal way.

- A feature that distinguishes [HMP92]'s semantic model is that it is *weakly monotonic*. This introduces some peculiarities, such as a predicate which can be both true and false for some timestamp value τ, but still at different positions in the timed trace. In other words, weak monotonicity really exposes the existence of "two times", one being the position in the timed word, the other being the value of the timestamp.

  An analysis of digitization may therefore consider what are the features which are peculiar to weak monotonicity, and what would change by switching to a strongly monotonic time.

- More generally, we already discussed how digitization is an intrinsically *semantic* notion, while sampling invariance is defined for formulas. Indeed, [HMP92]'s analysis of digitizable formulas is limited to two simple classes of formulas, namely bounded response and bounded invariance properties, which basically represent the notion of upper and lower bound on a response, respectively.

  Therefore, we may extend the analysis of digitizability to seek a wider characterization of digitizable formulas. Conversely, we may also provide a semantic counterpart to sampling invariance, and carry out the comparison in a semantic setting.

- A feature of our framework is that it changes interval bounds in formulas, in particular by *weakening formulas* when passing from discrete to continuous in order to preserve entailment. [HMP92] also introduces a notion of weakening (and strengthening) of formulas, although it uses it for different purposes (namely to outline verification techniques).

  We may see if [HMP92]'s notion of weakening can be used in a similar manner as our adaptation, and compare the resulting framework.

- Notice that digitization always rounds timestamp values to the nearest *integer value*; in other words, it approximates to within a precision of one. It would be straightforward, though, to generalize the definition to other fixed values, with a finer grain than one unit. This has been done in other works (see Related Works in [FR05]). It seems that such changes would not affect our comparison in any "qualitative" way, as changing this just amount to a rescaling of timing informations.

- More generally, we notice that the timed trace semantics has *several peculiar* (and, somewhat, unintuitive [HR04]) *features* that make a comparison with other semantic models more difficult and problematic. Among other things (and see Section 6 for references):

  - metric temporal logics over timed trace models are strictly *less expressive* than over interval-based sequences; therefore, one may suggest to restrict comparisons to properties expressible in both models;
  - a straightforward way to describe a timed trace as an interval-based sequence is to introduce *isolated events* at any timestamp, separated by "no-action" intervals where all predicates are false; notice that such sequences would all fail to satisfy the regularity constraint χ and thus would be trivially sampling invariant;
  - otherwise, one could represent interval-based sequences as timed traces with "samplings" taken to within some precision (and, possibly, according to the property at hand). Notice in particular that the granularity of such "samplings" would depend in general on the constants used in the formulas, if one seeks to preserve entailment. Possibly, the granularity should also be related to the regularity of the interval-based sequence (as mandated by χ).

4 Formalizing Timed Automata in $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO

It is well-known from the literature that automata languages cannot be defined using "standard" propositional temporal logics [Wol83], as the latter lack the ability to "count", which is instead a feature of finite-state automata [MP72]. In particular, more recent work has shown that there are languages accepted by timed automata [AD94] that cannot be defined by any metric temporal logic formula. See Section 6 for more extensive references about these facts.

This expressiveness gap notwithstanding, it is still possible to pursue a *dual language* approach [FMMR06, FMM94] for automata and temporal logic. This consists basically in describing a system through a combination of an operational formalism and a descriptive formalism: in our case, timed automata are the operational formalism and metric temporal logics (and $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO in particular) are its descriptive counterpart. The dual language approach then requires to formalize the behavior of the automata through logic formulas, so that one can prove properties about the automaton by logic deduction (rather than, e.g.,

algorithmic methods). Moreover, if we have a formalization of timed automata in $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO, it is possible to exploit a notion of discretization on the description of the automata, by applying the results about sampling invariance to the formalization. This section shows how to formalize the behavior of timed automata in $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO.

We stress the fact that the aforementioned expressiveness results about "counting" do not prevent one from performing such a formalization: even if, in general, one cannot define the language accepted by a timed automaton through $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formulas, it is still possible to characterize the *runs* of the automaton through logic formulas. That is, one includes in the formalization the state of the automaton, and how it evolves in response to inputs.[14]

4.1 Timed Automata Definition

Let us recall the definition of timed automata; all of them are from the original paper by Alur and Dill [AD94].

4.1.1 Syntax of Timed Automata

First, we consider the definition of *clock constraints*, then we define timed automata themselves.

Clock constraints. For a set X of clock variables, the set $\Phi(X)$ of *clock constraints* ξ is defined inductively by

$$\xi ::= x \leq c \mid c \leq x \mid \neg\xi \mid \xi_1 \wedge \xi_2$$

where x is a clock in X and c is a constant in $\mathbb{Q}$.

Timed automaton. A *timed automaton* A is a tuple $\langle \Sigma, S, S_0, C, E \rangle$, where:

- Σ is a finite alphabet,
- $S = \{s_0, s_1, \ldots, s_{N_s-1}\}$ is a finite set of N_s states,
- $S_0 \subseteq S$ is a finite set of start states,
- C is a finite set of clocks, and
- $E \subseteq S \times S \times \Sigma \times 2^C \times \Phi(C)$ gives the set of transitions.
  An edge $\langle s, s', a, \Lambda, \xi \rangle$ represents a transition from state s to state s' on input symbol a. The set $\Lambda \subseteq C$ gives the clocks to be reset with this transition, and ξ is a clock constraint over C.

[14] We remark the fact that one could then switch to the formalization of the accepted language itself if the logic had a *projection* (a.k.a. hiding or existential quantification) operator (see Section 6), which is equivalent to saying that "a logic with projection can count". $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO is not endowed with projection, and therefore we will limit ourselves to the formalization of automata runs.

Simplifications. For simplicity, let us restrict all the automata we consider to avoid self-loops among the transitions; i.e., $\langle s, s, a, \Lambda, \xi \rangle \notin E$ for all $s \in S$. For our purposes, this restriction is without loss of generality, as it can be shown that any timed automaton with self-loops can be transformed into one that recognizes the same language but does not use any self-loops. Moreover, let us avoid the use of ϵ-moves, which can however be rendered by augmenting the input alphabet by a special symbol.

4.1.2 Semantics of Timed Automata

In order to define the semantics of timed automata, we recall the definition of timed words and timed languages. Notice that we have some differences with respect to the definition in Section 3.1; in the remainder we will reference to the following definitions only.

Timed words. A *timed word* over an alphabet Σ is a pair of sequences (σ, τ). σ is an infinite word $\sigma = \sigma_1\sigma_2\sigma_3\cdots$ over Σ. τ is a *time sequence*, that is an infinite sequence $\tau = \tau_1\tau_2\tau_3\cdots$ of time values $\tau_i \in \mathbb{R}_{\geq 0}$; the sequence must be strictly monotonic (that is $\tau_i < \tau_{i+1}$ for all $i \geq 1$) and divergent (that is, for all $r \in \mathbb{R}_{\geq 0}$ there exists a $i \geq 1$ such that $\tau_i > r$).

Timed languages. A *timed language* is defined as a set of timed words.

Run of a timed automaton. To define the semantics of timed automata as acceptors of timed languages, we introduce the notion of *run* of a timed automaton. A *run* of a timed automaton $\langle \Sigma, S, S_0, C, E \rangle$ over a timed word (σ, τ) is an infinite sequence of the form:

$$(s_0, \nu_0) \xrightarrow{\sigma_1,\tau_1} (s_1, \nu_1) \xrightarrow{\sigma_2,\tau_2} (s_2, \nu_2) \xrightarrow{\sigma_3,\tau_3} \cdots$$

where $s_i \in S$ and $\nu_i \in [C \to \mathbb{R}]$, for all $i \geq 0$, satisfying the following requirements:

- *Initiation*: $s_0 \in S_0$ and $\nu_0(x) = 0$ for all $x \in C$.
- *Consecution*: for all $i \geq 1$, there is an edge in E of the form $(s_{i-1}, s_i, \sigma_1, \Lambda_i, \xi_i)$ such that $\nu_{i-1}+\tau_i-\tau_{i-1}$ satisfies ξ_i and ν_i equals $[\Lambda_i \mapsto 0](\nu_{i-1}+\tau_i-\tau_{i-1})$.

Language accepted by a timed automaton. Consequently, given a timed automaton A, and a set $F \subseteq S$ of accepting states, we say that A accepts the timed word (σ, τ) iff the timed word has a run which visits some state in F infinitely often (Büchi acceptance condition). Finally, a timed automaton A accepts the language identified by all and only timed words accepted by A.

4.1.3 Berkeley Behaviors for Timed Automata

Since the ultimate goal of this section is to provide a formalization that allows us to harness the discretization results about sampling invariance over languages defined by timed automata, we would like not only to formalize timed automata in $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO, but also to formalize them in a way which complies with the requirements of sampling invariance. Namely, we seek a formalization over slowly-variable behaviors (according to χ), and with nesting-free formulas.

Under this respect, the present subsection extends the notion of non-Berkeley behaviors to timed automata. This allows us to identify exactly those timed words that are representable by non-Berkeley behaviors.

Zeno behaviors and divergence. First of all, let us recall that Zeno behaviors are those where the time values in τ, although ever increasing, converge to a finite value $\overline{\tau}$: time "stops" at the value $\overline{\tau}$. The requirement that timestamps sequences in timed words be *divergent* is assumed precisely to rule out Zeno timed words.

Berkeley behaviors and finite difference time. As shown in [AD94, Example 3.16], and first explicitly discussed in [CHR02], in timed automata "infinitely fast" Berkeley behaviors can manifest themselves even if we rule out Zeno phenomena. Let us discuss an example of this fact with some detail.

An example of Berkeley timed automaton. Let us consider the timed automaton of Figure 9, where $k \in \mathbb{R}_{>0}$ is some positive constant. Let $x_0 = 0$ and

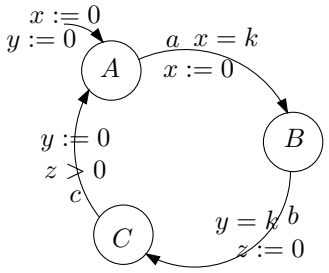

Figure 9: A timed automaton with Berkeley behavior.

$y_0 = 0$ be the initial values of the clocks x and y. Then, for all $i \geq 1$, let x_i be the values of the same clocks when entering the state A after the ith iteration of the loop. Let $\alpha_i, \beta_i, \gamma_i$ be the times spent sitting in the states A, B, C, respectively, during the ith iteration of the loop. This notation corresponds to the timed

word:

$$(a,\alpha_1)\,(b,\alpha_1+\beta_1)\,(c,\alpha_1+\beta_1+\gamma_1)\,\cdots$$
$$(a,\alpha_n+\sum_{i=1}^{n-1}(\alpha_i+\beta_i+\gamma_i))\,(b,\alpha_n+\beta_n+\sum_{i=1}^{n-1}(\alpha_i+\beta_i+\gamma_i))$$
$$(c,\alpha_n+\beta_n+\gamma_n+\sum_{i=1}^{n-1}(\alpha_i+\beta_i+\gamma_i))\,\cdots$$

Let us notice immediately that it must be $x_i+\alpha_{i+1}=k$, because of the guard on transition a, that is $k-\alpha_{i+1}=x_i$. Now, because of the guard on transition b, it must also be $\alpha_{i+1}+\beta_{i+1}=k$ (notice that y is reset when entering A). Finally, we have $x_{i+1}=\beta_{i+1}+\gamma_{i+1}$, since the clock x is last reset when entering B.

Thus, from the last equation, we get $x_{i+1}=\beta_{i+1}+\gamma_{i+1}=(k-\alpha_{i+1})+\gamma_{i+1}=x_i+\gamma_{i+1}$. This holds for all i, thus we can iteratively substitute and get: $x_{i+1}=x_i+\gamma_{i+1}=(x_{i-1}+\gamma_i)+\gamma_{i+1}=\cdots=x_0+\sum_{j=1}^{i+1}\gamma_j=\sum_{j=1}^{i+1}\gamma_j$.

Finally, let us notice that it must be $x_i\leq k$ for all i, otherwise the a transition cannot be taken. So we must have $\sum_{j=1}^{\infty}\gamma_j\leq k$, that is the residence times γ_i must get arbitrarily close to zero, while not being zero, in order to satisfy the constraint on the z clock.

Berkeley timed words and languages. Such kind of behaviors correspond to timed words where the difference between adjacent timestamps get arbitrarily small, even if time does not accumulate and stop. More precisely, we call *Berkeley* a timed word $(\sigma,\tau)=(\sigma_1,\tau_1)(\sigma_2,\tau_2)\cdots$ where the lower bound of the differences between adjacent timestamps $\lambda=\inf_{i\geq 1}(\tau_{i+1}-\tau_i)$ is zero, and *non-Berkeley* if it equals instead some $\theta\in\mathbb{R}_{>0}$. Similarly, a set of timed words $(\sigma,\tau)_{i\in I}$ (where $I\subseteq\mathbb{N}$) is Berkeley iff the lower bound $\inf_{i\in I}\lambda_i$ of the lower bounds of the timed words is zero. Consequently, a timed automaton is Berkeley iff the timed language it accepts is Berkeley.

In the rest of this section, when discussing the correspondence between timed words and timed behaviors in $^{\mathbb{R}}_{\mathbb{Z}}$TRIO, we assume that all timed languages are non-Berkeley, unless otherwise explicitly stated. We leave to future work the analysis of how ruling out Berkeley behaviors impacts on the expressiveness of timed automata.

4.2 Axiomatization of Timed Automata with $^{\mathbb{R}}_{\mathbb{Z}}$TRIO

Let us now consider how to describe in $^{\mathbb{R}}_{\mathbb{Z}}$TRIO all the runs of a given timed automaton. For simplicity, let us not consider acceptance conditions, that is let us assume that all states are accepting. Note, however, that introducing acceptance conditions (e.g., Büchi, Muller, etc) would be routine.

First of all, since timed automata are interpreted over timed words, whereas $^{\mathbb{R}}_{\mathbb{Z}}$TRIO formulas are interpreted over behaviors $b:\mathbb{T}\to D$, we have to relate the

two semantic models. Remember that we want to be able to consider *constrained* behaviors, i.e., regular according to the constraint $\chi_\circ$ for some fixed sampling period δ. Therefore, let us pick $\delta < \theta/2$, where θ is the lower bound of the transition times for the automaton. The reason for taking θ halved will be clear in the remainder.

4.2.1 Timed Words and Timed Behaviors

Let us consider a timed automaton $A = \langle \Sigma, S, S_0, C, E \rangle$; it is interpreted over timed words of the form $\langle \sigma, \tau \rangle$, where $\sigma \subseteq \Sigma^\omega$.

To map runs (over timed words) to behaviors, we have first to choose a set of basic *time-dependent items* for the $^{\mathbb{R}}_{\mathbb{Z}}$TRIO description. More precisely, we consider the following items.

- a time-dependent item st taking values to the set of states S;
- a time-dependent item in taking values to the alphabet set $\Sigma \cup \{\epsilon\}$;
- a time-dependent predicate $\mathsf{rs}()$ having the set of clocks C as domain;
- a time-dependent (Boolean) time-dependent predicate start.

Here it is the intuitive meaning of these items.

- $\mathsf{st} = s_i$ means that the automaton is in state $s_i \in S$;
- $\mathsf{in} = \sigma$ means that the symbol $\sigma \in \Sigma$ is currently being inputted (or no current input, in the case of ϵ);
- $\mathsf{rs}(x)$ means that the clock $x \in C$ is currently being reset;
- start represents the initialization event.

Note that this choice of items prevents multiple input symbols from arriving at the same time, whereas multiple resets (for different clocks) can happen at the same instant.

Mapping a timed word onto a timed behavior. Let us consider a run of the automaton A:

$$(s_0, \nu_0) \xrightarrow{\sigma_1, \tau_1} (s_1, \nu_1) \xrightarrow{\sigma_2, \tau_2} (s_2, \nu_2) \xrightarrow{\sigma_3, \tau_3} \cdots$$

Given a sampling period δ, we associate to the run the behavior b for the above items as follows.

- for every transition $\xrightarrow{\sigma_i, \tau_i} (s_i, \nu_i) \xrightarrow{\sigma_{i+1}, \tau_{i+1}}$, the item st takes the value s_i exactly from time τ_i to time τ_{i+1}, left-continuously;
- for every input pair (σ_i, τ_i), the item in takes the value σ_i exactly from time τ_i (included) to time $\tau_i + \delta$ (included); otherwise (i.e., at any other time) in takes the value ϵ;

- for every clock $x \in C$ that is reset when taking the transition corresponding to the input pair (σ_i, τ_i), the predicate $\mathsf{rs}(x)$ is true exactly from time τ_i (included) to time $\tau_i + \delta$ (included); moreover, $\mathsf{rs}(x)$ holds from time 0 until δ (included) for every clock $x \in C$;
- start is true from time $-\infty$ to time δ (included), and false everywhere else.

Mapping a timed behavior onto a timed word. The inverse mapping can be defined, for the present purposes, as follows. Simply, given a timed behavior b, if there exists a timed word (σ, τ) such that the latter maps to the former according to the previously defined mapping, then we say that b maps to (σ, τ), otherwise the mapping of b is undefined. Notice that, when it is defined, this inverse mapping is also uniquely defined.

4.2.2 Axiomatization of a Timed Automaton

Let us finally consider a timed automaton $A = \langle \Sigma, S, S_0, C, E \rangle$. We now give a set of $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO axioms that formally describe its runs as behaviors, according to the mapping we outlined above. First, we present the axioms; afterward we justify their correctness.

In the remainder, we assume that there are no edges going from a state to itself (self-loops); this does not impacts on the expressiveness of timed automata, as it can be shown.

Translating clock constraints into formulas. We associate a $\frac{\mathbb{R}}{\mathbb{Z}}$TRIO formula Ξ to every clock constraint ξ. In order to simplify this step of the formalization, we assume that all constants in the clock constraints are strictly greater than δ. It can be shown that this can always be achieved by properly scaling all constants by some suitable factor [AD94]; this corresponds to a change in the time scale. This restriction is inessential in the formalization, but we introduce it for the sake of simplicity, leaving to future work the full discussion of all the possible cases.

Let us define inductively Ξ on the structure of ξ as follows:

$$\begin{array}{lcl} x \leq c & \longrightarrow & \mathrm{WithinP}_{\mathrm{ei}}(\mathsf{rs}(x), c - \delta) \\ c \leq x & \longrightarrow & \mathrm{Lasted}_{\mathrm{ee}}(\neg\mathsf{rs}(x), c - \delta) \\ \neg\xi & \longrightarrow & \neg\Xi \\ \xi_1 \wedge \xi_2 & \longrightarrow & \Xi_1 \wedge \Xi_2 \end{array}$$

Basically, Ξ translates the guard ξ by comparing the current time to the last time a reset for the clock x happened. The offset δ is introduced to take into account the fact that each reset item holds continuously for δ time units from the moment when the reset is actually triggered (to comply with the slow variability requirement $\chi_\circ$).[15]

[15] It can be shown that derived properties of composite clock constraints hold for the corresponding temporal logic formulas. For instance $x \leq x \wedge c \leq x$ corresponds to

Necessary conditions for state change. Let us state the necessary conditions that characterize a state change. For any pair of states $s_i, s_j \in S$ such that $\langle s_i, s_j, \sigma, \Lambda, \xi \rangle \in E$ for some σ, $\Lambda = c_1, \ldots, c_r$, ξ, we introduce the axiom:[16]

$$\begin{aligned} &\mathrm{UpToNow}(\mathsf{st} = s_i) \wedge \mathrm{NowOn}(\mathsf{st} = s_j) \\ &\qquad \Rightarrow \quad \mathrm{NowOn}(\mathsf{in} = \sigma) \wedge \mathrm{NowOn}(\mathsf{rs}(c_1) \wedge \cdots \wedge \mathsf{rs}(c_r)) \wedge \Xi \end{aligned} \tag{12}$$

Sufficient conditions for state change. We have multiple sufficient condition for state changes; basically, they account for reactions to reading input symbols and resetting clocks. Let us consider input first: for each input symbol $\sigma \in \Sigma$, let us consider all the relations of the form $\langle s_i^k, s_j^k, \sigma, \Lambda^k, \xi^k \rangle \in E$. Then, we introduce the axiom:

$$\begin{aligned} &\mathrm{NowOn}(\mathsf{in} = \sigma) \Rightarrow \\ &\bigvee_k \left(\mathrm{UpToNow}\big(\mathsf{st} = s_i^k\big) \wedge \mathrm{NowOn}\big(\mathsf{st} = s_j^k\big) \wedge \mathrm{NowOn}\left(\bigwedge_{\lambda \in \Lambda^k} \mathsf{rs}(\lambda) \right) \wedge \Xi^k \right) \end{aligned} \tag{13}$$

This postulates the fact that, whenever a symbol σ is inputted, it arrives under conditions that are compatible with one of the transitions specified by the relation E.

Similarly, for each reset of a clock $c \in C$, let us consider all the relations of the form $\langle s_i^k, s_j^k, \sigma^k, \Lambda^k, \xi^k \rangle \in E$, such that $c \in \Lambda^k$ for all k. Then, we introduce the axiom:

$$\begin{aligned} &\mathrm{NowOn}(\mathsf{rs}(c)) \Rightarrow \\ &\qquad \bigvee_k \left(\mathrm{UpToNow}\big(\mathsf{st} = s_i^k\big) \wedge \mathrm{NowOn}\big(\mathsf{st} = s_j^k\big) \wedge \mathrm{NowOn}\big(\mathsf{in} = \sigma^k\big) \right. \\ &\qquad\qquad \left. \wedge \mathrm{NowOn}\left(\bigwedge_{\lambda \in \Lambda^k} \mathsf{rs}(\lambda) \right) \wedge \Xi^k \right) \vee \mathrm{NowOn}(\mathsf{start}) \end{aligned} \tag{14}$$

Initialization and liveness condition. We complete our axiomatization by first describing the system initialization: at the beginning the predicate start is

$\phi = \mathrm{WithinP}_{\mathrm{ei}}(\mathsf{rs}(x), c - \delta) \wedge \mathrm{Lasted}_{\mathrm{ee}}(\neg \mathsf{rs}(x), c - \delta)$; it can be shown that if ϕ holds, then $\mathsf{rs}(x)$ last held exactly $c - \delta$ time units in the past. Therefore, the last reset has occurred exactly $c - \delta + \delta = c$ time units in the past, which corresponds to the condition $x = c$ being true, as expected.

[16] Throughout the formalization, we assume $\mathrm{NowOn}(p) = \mathrm{Lasts}_{\mathrm{ee}}(p, \delta)$, and $\mathrm{UpToNow}(p) = \mathrm{Lasted}_{\mathrm{ee}}(p, \delta)$.

true for δ time units, the system is in an initial state and all the clock are reset.

$$\begin{aligned}\mathrm{UpToNow}(\mathsf{start}) \wedge \mathrm{NowOn}(\neg\mathsf{start}) \\ \Rightarrow \quad \mathrm{AlwP}(\mathsf{in} = \epsilon) \wedge (\exists s \in S_0 : \mathrm{AlwP}(\mathsf{st} = s)) \\ \wedge (\forall \lambda \in C : \mathrm{AlwP}(\mathsf{rs}(\lambda))) \end{aligned} \quad (15)$$

Then, we have to say that start actually becomes true, sometimes, and it has been true always in the past and will be false in the future.

$$\begin{aligned}\mathrm{Som}(\mathrm{AlwP}(\mathsf{start})) \wedge \mathrm{Som}(\neg\mathsf{start}) \\ \wedge (\mathrm{AlwP}(\mathsf{start}) \wedge \mathrm{NowOn}(\neg\mathsf{start}) \Rightarrow \mathrm{AlwF}(\neg\mathsf{start})) \end{aligned} \quad (16)$$

Recall that the nesting of the temporal operator Som and AlwP does not affect sampling invariance of the resulting formula.

Finally, a "liveness" condition states that we eventually have to move out of every state (this corresponds to the fact that timed traces are made of infinite words). Thus, for every state $s_i \in S$, let $S_i \subset S$ be the set of states that are directly reachable from s_i through a single transition; then we consider the axiom:

$$\mathrm{UpToNow}(\mathsf{st} = s_i) \Rightarrow \mathrm{SomF}\left(\bigvee_{s \in S_i} \mathsf{st} = s\right) \quad (17)$$

4.2.3 Correctness and Completeness of the Axiomatization

Let us now sketch a proof of the correctness and completeness of the above axiomatization of timed automata.

Completeness of the axiomatization. In order to show the completeness of the above axiomatization, let us consider a generic run of a given timed automaton, for which the axioms introduced in Section 4 have been spelled out.

$$(s_0, \nu_0) \xrightarrow{\sigma_1, \tau_1} (s_1, \nu_1) \xrightarrow{\sigma_2, \tau_2} (s_2, \nu_2) \xrightarrow{\sigma_3, \tau_3} \cdots$$

We have to show that all the above axioms are made true by the timed behavior associated with the given timed word. First of all, let us notice that axiom 16 is true by construction, if we consider the behavior of the start predicate associated to any behavior. Considering the other axioms, since they all are implicitly universally quantified with respect to time (as every TRIO formula), we prove their truth by induction of the length of the timed word.

Base case: up to time τ_1. Without loss of generality, we assume that $\tau_1 > \delta$. Recall that:

- start is true from $-\infty$ to δ;
- $s_0 \in S_0$ is true from 0 to $\tau_1 > \delta$;

- $\mathsf{in} = \epsilon$ holds from 0 to τ_1;
- $\mathsf{rs}(c)$ holds from 0 to δ for all clocks $c \in C$.

As a consequence, all axioms (but axiom 14) are trivially true from $-\infty$ until δ excluded, as all their antecedents are false (since they consider an interval of length δ with their UpToNow operators). For what concerns axiom 14, it is also true, as at time 0 NowOn(start) holds.

Now, from time δ up to time τ_1 excluded, we have that:

- start is false;
- $s_0 \in S_0$ is true;
- $\mathsf{in} = \epsilon$ holds;
- $\mathsf{rs}(c)$ is false for all clocks $c \in C$.

Consequently, axioms 12–14 are trivially true since their antecedents are false. Axiom 15 is also true, since after δ its antecedent is false, whereas exactly at δ its consequent is satisfied by construction. Regarding axiom 17, it is also true as the state will change to s_1 at τ_1.

Inductive case: from time τ_i to time τ_{i+1} . We now show that, assuming the axioms are satisfied until time τ_i excluded, then they are also satisfied until time τ_{i+1} excluded. Let us first consider what happens exactly at time τ_i. Let $\langle s_{i-1}, s_i, \sigma_i, \Lambda_i, \xi_i \rangle \in E$ be the transition that is taken; therefore:

- start is false;
- UpToNow($\mathsf{st} = s_{i-1}$) is true;
- NowOn($\mathsf{st} = s_i$) is true;
- $\mathsf{in} = \sigma_i$ holds from τ_i to $\tau_i + \delta$;
- $\mathsf{rs}(c)$ holds from τ_i to $\tau_i + \delta$ for all clocks $c \in \Lambda_i$.

Therefore, axiom 12 holds, as the symbol σ_i is inputed, the right clocks are reset and the condition Ξ is met by construction, thus making true both sides of the implication. Similarly for axioms 13 and 14, both sides of their implications are true. Axiom 15 holds trivially as start is false, whereas axiom 17 is satisfied for $s = s_i$.

From time τ_i until time τ_{i+1}, we have that:

- start is false;
- s_i is true;
- $\mathsf{in} = \sigma_i$ holds until $\tau_i + \delta$, and $\mathsf{in} = \epsilon$ holds afterward;
- $\mathsf{rs}(c)$ for all clocks $c \in \Lambda_i$ holds until $\tau_i + \delta$, and $\mathsf{rs}(c)$ is false for all clocks $c \in C$ afterward.

Thus, all axioms are trivially true, their antecedents being false, except for axiom 17, which is however satisfied for $s = s_{i+1}$.

Correctness of the axiomatization. Conversely, in order to assess the correctness of the axiomatization, we have to show that any timed behavior satisfying all the axioms corresponds to a run which is a valid run of the automaton. This is also shown by induction on the timed behavior.

Base case: up to time τ_0. Let us first of all consider axiom 16. Let t_s be the time instant at which AlwP(start) holds. Because of the condition $\chi_\circ$, it must be $t_s \geq \delta$. Moreover, axiom 16 also implies that start cannot be always true, but it must exist an instant $\tau_0 \geq t_s$ where start changes its value from true to false; more precisely, still because of the regularity conditions, it must be UpToNow(start) and NowOn($\neg\mathsf{start}$) at τ_0.

Let us consider axiom 15 at τ_0. Since AlwP($\mathsf{st} = s$), AlwP($\mathsf{in} = \epsilon$), and AlwP($\mathsf{rs}(\lambda)$) for all clocks λ, the value of the state, input, and resets is completely defined until τ_0. Therefore, we realize that this corresponds to the first element $(s, 0)$ of a valid timed word, which just specified the reset conditions of the automaton.

Inductive case: from time τ_i to time τ_{i+1}. The state s_i holds until time τ_i by inductive hypothesis. It cannot be that $\mathsf{st} = s_i$ holds forever, because of axiom 17. More precisely, thanks to the regularity of the behaviors, it is well defined a time instant $\tau_{i+1} > \tau_i + \delta > \tau_i$ such that st changes its value from s_i to some s_{i+1} at τ_{i+1}. Let us now trace the run of the automaton from time τ_i to τ_{i+1}.

Let us first notice that start is constantly false, as it has been so since time τ_0, due to axiom 16.

Exactly at time τ_i, it might be that NowOn($\mathsf{in} = \sigma$) for some input character σ. This would correspondingly set the resets $\mathsf{rs}(\lambda)$ for the appropriate clocks λ's; this behavior would of course be compatible with both axioms 12 and 14. Otherwise, if NowOn(ϵ) at τ_i, notice that there would be no resets from time τ_i to time $\tau_i + \delta$.

Now, notice that in must equal ϵ for all remaining time instants, up to τ_{i+1}. In fact, if this would not be the case, then we should record a state transition at that time, because of axiom 13. Similarly, by axiom 14, $\mathsf{rs}(c)$ must be false for all clocks c, otherwise a state transition should also occur (as we already noted, NowOn(start) is false).

Correspondingly, we can add the next step in the timed word $(s_{i-1}, \nu_{i-1}) \xrightarrow{\sigma, \tau_i} (s_i, \nu_i)$, compatibly with the timed behavior of the axiomatized automaton.

4.2.4 Axiomatization of Berkeley Timed Automata

Let us also discuss how it is possible to modify the above axiomatization in order to describe the behavior of any non-Zeno timed automaton, including those with Berkeley behaviors. Notice that now we forget about the requirements of the integration framework (such as the non-Berkeleyness constraint $\chi_\circ$), and we only focus to formalizing the behavior of a *generic* timed automaton with

$^{\mathbb{R}}_{\mathbb{Z}}$TRIO. In this case, we give an extension of the dual language approach to deal with any timed automaton, giving up the possibility of applying our notion of discretization through sampling invariance.

To this end, it is convenient to declare the item st as a *state*, whereas all the other items are *events* [GM01]. Note that the input item would now simply be false for all $\sigma \in \Sigma$ when no character is inputted. Then, these items would be true exactly at those times when an input is received, a reset is triggered, or the system is started.

The translation of clock constraints into formulas would take into account these changes, so that the bounds of the WithinP and of the Lasted would simply be the constant c against which a clock is compared, without the δ offset.

Finally, the overall structure of the axioms would be unchanged, except that all the references to event items would now get rid of the UpToNow or NowOn operators. On the contrary, UpToNow and NowOn remains when their arguments are states; however we now have to use the $\epsilon\widetilde{\text{UpToNow}}$ and $\epsilon\widetilde{\text{NowOn}}$ versions (defined in Section 1.1), as states are no more guaranteed to hold over δ. For clarity, we replicate here the new versions of the axioms.

$$\begin{aligned}\text{UpToNow}(\mathsf{st} = s_i) \wedge \text{NowOn}(\mathsf{st} = s_j) \\ \Rightarrow \quad \mathsf{in} = \sigma \wedge (\mathsf{rs}(c_1) \wedge \cdots \wedge \mathsf{rs}(c_r)) \wedge \Xi\end{aligned} \tag{18}$$

$$\mathsf{in} = \sigma \Rightarrow \bigvee_k \left(\text{UpToNow}\big(\mathsf{st} = s_i^k\big) \wedge \text{NowOn}\big(\mathsf{st} = s_j^k\big) \wedge \left(\bigwedge_{\lambda \in \Lambda^k} \mathsf{rs}(\lambda) \right) \wedge \Xi^k \right) \tag{19}$$

$$\begin{aligned}\mathsf{rs}(c) \Rightarrow \bigvee_k \Bigg(\text{UpToNow}\big(\mathsf{st} = s_i^k\big) \wedge \text{NowOn}\big(\mathsf{st} = s_j^k\big) \wedge \mathsf{in} = \sigma^k \\ \wedge \left(\bigwedge_{\lambda \in \Lambda^k} \mathsf{rs}(\lambda) \right) \wedge \Xi^k \Bigg) \vee \mathsf{start}\end{aligned} \tag{20}$$

$$\begin{aligned}\mathsf{start} \Rightarrow \quad \text{AlwP}(\forall \sigma \in \Sigma : \mathsf{in} \neq \sigma) \wedge (\exists s \in S_0 : \text{AlwP}(\mathsf{st} = s)) \\ \wedge (\forall \lambda \in C : \mathsf{rs}(\lambda))\end{aligned} \tag{21}$$

$$\text{Som}(\mathsf{start}) \wedge (\mathsf{start} \Rightarrow \text{AlwP}(\neg \mathsf{start}) \wedge \text{AlwF}(\neg \mathsf{start}))$$

$$\mathrm{UpToNow}(\mathsf{st} = s_i) \Rightarrow \mathrm{SomF}\left(\bigvee_{s \in S_i} \mathsf{st} = s\right) \tag{22}$$

Now, note that the above correctness and completeness proofs hold, with some technical modifications, for the unconstrained case as well. For brevity, we omit the details.

4.3 An Example

Let us apply the above axiomatization on a simple example: the automaton of Figure 10, modeling a train. In the model, the input symbols a, i, o, e represent, respectively, the train approaching a crossing area, entering the area, exiting it, and notifying its exit. The example automaton is taken from [AD94].

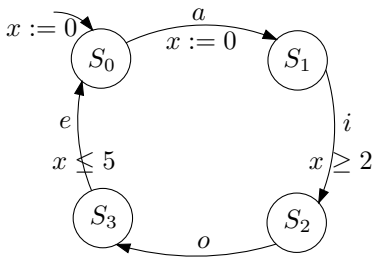

Figure 10: A timed automaton modeling a train.

It is straightforward to realize that the automaton accepts the language:

$$\{((aioe)^\omega, \tau) \mid \forall i (i \equiv 1 \bmod 4 \rightarrow \tau_{i+1} \geq \tau_i + 2 \wedge \tau_{i+3} \leq \tau_i + 5)\}$$

Let us now write out all the axioms. First of all, axiom 12 is implemented as the four following axioms.

$$\mathrm{UpToNow}(\mathsf{st} = S_0) \wedge \mathrm{NowOn}(\mathsf{st} = S_1) \Rightarrow \mathrm{NowOn}(\mathsf{in} = a) \wedge \mathrm{NowOn}(\mathsf{rs}(x))$$
$$\mathrm{UpToNow}(\mathsf{st} = S_1) \wedge \mathrm{NowOn}(\mathsf{st} = S_2) \Rightarrow \mathrm{NowOn}(\mathsf{in} = i) \wedge \mathrm{Lasted}_{\mathrm{ee}}(\neg\mathsf{rs}(x), 2 - \delta)$$
$$\mathrm{UpToNow}(\mathsf{st} = S_2) \wedge \mathrm{NowOn}(\mathsf{st} = S_3) \Rightarrow \mathrm{NowOn}(\mathsf{in} = o)$$
$$\mathrm{UpToNow}(\mathsf{st} = S_3) \wedge \mathrm{NowOn}(\mathsf{st} = S_0) \Rightarrow \mathrm{NowOn}(\mathsf{in} = e) \wedge \mathrm{WithinP}_{\mathrm{ii}}(\mathsf{rs}(x), 5 - \delta)$$

Let us now consider axiom 13, which gets to:

$$\mathrm{NowOn}(\mathsf{in} = a) \Rightarrow \mathrm{UpToNow}(\mathsf{st} = S_0) \wedge \mathrm{NowOn}(\mathsf{st} = S_1) \wedge \mathrm{NowOn}(\mathsf{rs}(x))$$
$$\mathrm{NowOn}(\mathsf{in} = i) \Rightarrow \mathrm{UpToNow}(\mathsf{st} = S_1) \wedge \mathrm{NowOn}(\mathsf{st} = S_2) \wedge \mathrm{Lasted}_{\mathrm{ee}}(\neg\mathsf{rs}(x), 2 - \delta)$$
$$\mathrm{NowOn}(\mathsf{in} = o) \Rightarrow \mathrm{UpToNow}(\mathsf{st} = S_2) \wedge \mathrm{NowOn}(\mathsf{st} = S_3)$$
$$\mathrm{NowOn}(\mathsf{in} = e) \Rightarrow \mathrm{UpToNow}(\mathsf{st} = S_3) \wedge \mathrm{NowOn}(\mathsf{st} = S_0) \wedge \mathrm{WithinP}_{\mathrm{ii}}(\mathsf{rs}(x), 5 - \delta)$$

Axiom 14 gets simply into:

$$\mathrm{NowOn}(\mathsf{rs}(x)) \Rightarrow (\mathrm{UpToNow}(\mathsf{st} = S_0) \wedge \mathrm{NowOn}(\mathsf{st} = S_1) \wedge \mathrm{NowOn}(\mathsf{in} = a)) \vee \mathrm{NowOn}(\mathsf{start})$$

We do not write out explicitly the initialization conditions of axioms 15–16, as they do not depend on the actual automaton (except for the simple condition on start states). Thus, let us just consider the liveness condition of axiom 17; this is rendered by the following four axioms.

$$\mathrm{UpToNow}(\mathsf{st} = S_0) \Rightarrow \mathrm{SomF}(S_1)$$
$$\mathrm{UpToNow}(\mathsf{st} = S_1) \Rightarrow \mathrm{SomF}(S_2)$$
$$\mathrm{UpToNow}(\mathsf{st} = S_2) \Rightarrow \mathrm{SomF}(S_3)$$
$$\mathrm{UpToNow}(\mathsf{st} = S_3) \Rightarrow \mathrm{SomF}(S_0)$$

This concludes the axiomatization for the present example.

5 An Example of Integration: The Controlled Reservoir

Let us now discuss an example that demonstrates our approach to integration. Let us consider a simple control system made of two modules: a reservoir and a controller. We model the former using continuous-time TRIO, and the latter using discrete-time TRIO. This mirrors the assumptions that the reservoir models a physical systems, where quantities vary continuously, while the controller is a digital device, that updates its state at periodic instants.

5.1 System Specification

The reservoir can nondeterministically leak; this is modeled by the Boolean item L which is true iff the reservoir leaks. Leaking can happen at any time. The reservoir can also be filled with new liquid: this happens whenever the Boolean item F is true. It is the controller which is responsible for setting F to true, thus replacing the fluid that has been split, whenever needed. Finally, the reservoir has a (nonnegative) real-valued time-dependent item I that represents the measure of the level of fluid in the reservoir at any instant.

It is customary to constraint the items of a continuous-time specification so that they cannot take physically impossible behaviors such as discontinuities or Zeno behaviors [GM01]. In our example, we model the Boolean items F and L as *states*, that is, roughly speaking, as items with a piecewise-constant non-Zeno behavior.

When the reservoir is leaking, the level of fluid decreases at a constant rate of $\mathrm{r_l}$ units of fluid per unit of time. Conversely, when it is being filled, the level of fluid increases at a constant rate of $\mathrm{r_f}$, and we assume that $\mathrm{r_f} > \mathrm{r_l}$, i.e.,

the filling can contrast the leaking. The four possible resulting behaviors are described by the following TRIO axioms.

Axiom 1 (reservoir$^{\mathbb{R}}$.level_behavior_1).

$$\mathrm{Lasts}_{\mathrm{ee}}(\mathsf{F} \wedge \mathsf{L}, t) \wedge \mathsf{l} = l \;\Rightarrow\; \mathrm{Futr}(\mathsf{l} = l + (\mathrm{r_f} - \mathrm{r_l})t, t)$$

Axiom 2 (reservoir$^{\mathbb{R}}$.level_behavior_2).

$$\mathrm{Lasts}_{\mathrm{ee}}(\mathsf{F} \wedge \neg\mathsf{L}, t) \wedge \mathsf{l} = l \;\Rightarrow\; \mathrm{Futr}(\mathsf{l} = l + \mathrm{r_f}t, t)$$

Axiom 3 (reservoir$^{\mathbb{R}}$.level_behavior_3).

$$\mathrm{Lasts}_{\mathrm{ee}}(\neg\mathsf{F} \wedge \mathsf{L}, t) \wedge \mathsf{l} = l \;\Rightarrow\; \mathrm{Futr}(\mathsf{l} = \max(l - \mathrm{r_l}t, 0), t)$$

Axiom 4 (reservoir$^{\mathbb{R}}$.level_behavior_4).

$$\mathrm{Lasted}_{\mathrm{ee}}(\neg\mathsf{F} \wedge \neg\mathsf{L}, t) \wedge \mathsf{l} = l \;\Rightarrow\; \mathrm{Futr}(\mathsf{l} = l, t)$$

For specifying the control action of the controller we define two constant values $\mathrm{l}, \mathrm{t} \in \mathbb{R}_{\geq 0}$ that represent, respectively, the minimum desired level of fluid in the reservoir and a threshold below which the control action is triggered. More precisely, t is such that $\mathrm{t} > \mathrm{l} + \mathrm{r_l}\delta$, i.e., a leaking cannot empty the difference $\mathrm{t} - \mathrm{l}$ in less than δ time units, for some $\delta \in \mathbb{R}_0^+$. Thus, we formalize very simply control action as follows (remember that this is interpreted over discrete time):

Axiom 5 (controller$^{\mathbb{Z}}$.filling_behavior).

$$\mathsf{l} < \mathrm{t} \;\Rightarrow\; \mathsf{F}$$

We also assume that the reservoir is initially filled properly.

Axiom 6 (reservoir$^{\mathbb{R}}$.initialization).

$$\mathrm{Som}(\mathrm{AlwP_e}(\mathsf{l} \geq \mathrm{t}))$$

5.2 Sampling Invariant Derived Specification

Axioms 1–2 above use free variables, so they are not $^{\mathbb{R}}_{\mathbb{Z}}$TRIO formulas. However, it is straightforward to derive some simpler formulas, using only constant values, that are full $^{\mathbb{R}}_{\mathbb{Z}}$TRIO formulas and can therefore be shown to be sampling invariant. They are listed below as four theorems, whose elementary proofs are also shown for clarity.

Theorem 7 (reservoir$^{\mathbb{R}}$.abstract_behavior_1).

$$\mathrm{Lasts}_{\mathrm{ee}}(\mathsf{F}, \delta) \wedge \mathsf{l} \geq \mathrm{l} \;\Rightarrow\; \mathrm{Futr}(\mathsf{l} \geq \mathrm{l}, \delta)$$

Proof. Since L is a state item, it changes its value a finite number of times in the interval of length δ from the current instant. In each of the resulting constancy intervals, we have that either $\mathsf{F} \wedge \mathsf{L}$ or $\mathsf{F} \wedge \neg\mathsf{L}$ holds; in both cases the level of fluid must increase in the interval, by Axioms **levelLbehavior_1** and **levelLbehavior_2**, respectively. Thus, after a finite number of increases, it must be $\mathsf{l} > \mathrm{l} \geq \mathrm{l}$ after δ time instants. □

Theorem 8 (reservoir$^{\mathbb{R}}$.abstract_behavior_2).

$$\mathsf{l} \geq \mathrm{t} \;\Rightarrow\; \mathrm{Futr}(\mathsf{l} \geq \mathsf{l}, \delta)$$

Proof. Let l' be the value of l at δ time instants in the future, and l the current value. By Axioms 1 through 4, it is simple to understand that the level cannot go below the value $\max(\mathsf{l} - \mathrm{r_l}\delta, 0) = \mathsf{l} - \mathrm{r_l}\delta \geq \mathrm{l}$, since we are assuming that $\mathrm{t} - \mathrm{r_l}\delta > \mathrm{l}$. □

Theorem 9 (reservoir$^{\mathbb{R}}$.abstract_behavior_3).

$$\mathrm{Lasts}_{\mathrm{ee}}(\neg\mathsf{F} \wedge \neg\mathsf{L}, \delta) \wedge \mathsf{l} \geq \mathrm{l} \;\Rightarrow\; \mathrm{Futr}(\mathsf{l} \geq \mathrm{l}, \delta)$$

Proof. Let l' be the value of l at δ time instants in the future, and l the current value. Then, $\mathsf{l}' = \mathsf{l} \geq \mathrm{l}$ by Axiom **levelLbehavior_4**. □

Notice, instead, that the other axioms already qualify as $^{\mathbb{R}}_{\mathbb{Z}}$TRIO formulas.

5.3 System Requirement

The goal of the verification problem for this example is to prove the following $^{\mathbb{R}}_{\mathbb{Z}}$TRIO formula, which formalizes the requirement that the level of the reservoir never goes below the value l. Note that this formula can be equivalently interpreted over the reals or the integers.

Theorem 10 (levelLinvariance).

$$\mathrm{Alw}(\mathsf{l} \geq \mathrm{l})$$

5.4 Adapted Specification

Let us now put all the relevant formulas in normal form; note that we omit the parenthesis index in operators (such as the $\rangle$ in $\mathrm{Until}_{I\rangle}(\cdots)$), whenever it can be anything. For brevity, let us not describe the full transcription process.

Theorem 11 (reservoir$^{\mathbb{R}}$.abstract_behavior_1-normal).

$$\mathsf{l} < \mathrm{l} \;\vee\; \mathrm{Until}_{(0,\delta)}(\mathrm{true}, \neg\mathsf{F}) \;\vee\; \mathrm{Releases}_{[\delta,\delta]}(\mathrm{false}, \mathsf{l} \geq \mathrm{l})$$

Theorem 12 (reservoir$^{\mathbb{R}}$.abstract_behavior_2-normal).

$$\mathsf{l} < \mathrm{t} \;\vee\; \mathrm{Releases}_{[\delta,\delta]}(\mathrm{false}, \mathsf{l} \geq \mathrm{l})$$

Theorem 13 (reservoir$^{\mathbb{R}}$.abstract_behavior_3-normal).

$$\mathsf{I} < \mathrm{l} \ \vee \ \mathrm{Until}_{(0,\delta)}(\mathrm{true}, \mathsf{F} \vee \mathsf{L}) \ \vee \ \mathrm{Releases}_{[\delta,\delta]}(\mathrm{false}, \mathsf{I} \geq \mathrm{l})$$

In the following axiom, note that S is the auxiliary item introduced to eliminate nesting.

Axiom 14 (reservoir$^{\mathbb{R}}$.initialization-normal).

$$\big(\big(\mathsf{S} \wedge \mathrm{Released}_{(0,\infty)}(\mathrm{false}, \mathsf{I} \geq \mathrm{t})\big) \ \vee \ \big(\neg\mathsf{S} \wedge \mathrm{Since}_{(0,\infty)}(\mathrm{true}, \mathsf{I} < \mathrm{t})\big)\big) \\ \wedge \big(\mathrm{Since}_{(0,\infty)}(\mathrm{true}, \mathsf{S}) \vee \mathsf{S} \vee \mathrm{Until}_{(0,\infty)}(\mathrm{true}, \mathsf{S})\big)$$

Next, we adapt the continuous-time Theorems 11–13 and Axiom 14 into discrete-time formulas according to a sampling period δ. The resulting formulas, after putting them back into more readable form, are the following, where $\mathrm{Lasts}_{\mathrm{ii}}(\phi, t) = \forall d \in [0, t] : \mathrm{Dist}(\phi, d)$.

Theorem 15 (reservoir$^{\mathbb{Z}}$.abstract_behavior_1).

$$\mathrm{Lasts}_{\mathrm{ii}}(\mathsf{F}, 1) \wedge \mathsf{I} \geq \mathrm{l} \ \Rightarrow \ \mathrm{NowOn}(\mathsf{I} \geq \mathrm{l})$$

Theorem 16 (reservoir$^{\mathbb{Z}}$.abstract_behavior_2).

$$\mathsf{I} \geq \mathrm{t} \ \Rightarrow \ \mathrm{NowOn}(\mathsf{I} \geq \mathrm{l})$$

Theorem 17 (reservoir$^{\mathbb{Z}}$.abstract_behavior_3).

$$\mathrm{Lasts}_{\mathrm{ii}}(\neg\mathsf{F} \wedge \neg\mathsf{L}, 1) \wedge \mathsf{I} \geq \mathrm{l} \ \Rightarrow \ \mathrm{NowOn}(\mathsf{I} \geq \mathrm{l})$$

Axiom 18 (reservoir$^{\mathbb{Z}}$.initialization).

$$\mathrm{Som}(\mathrm{AlwP}_{\mathrm{i}}(\mathsf{I} \geq \mathrm{t}))$$

5.5 System Verification

Let us assume that the behavior constraint $\chi^{\phi}_{\bullet}(15-18)$ for Formulas 15–18 holds. In the following section we will discuss the actual impact of the constraint and how it can be removed.

Proof of Theorem 10 ($\mathrm{Alw}(\mathsf{I} \geq \mathrm{l})$*).* Thanks to Axiom 18, we have a time instant u at which $\mathrm{AlwP}_{\mathrm{i}}(\mathsf{I} \geq \mathrm{t})$ holds. This instant serves as the base of the induction.

So, for all instants up to u the property $\mathsf{I} \geq \mathrm{l}$ holds trivially.

Let us now consider a generic time instant $v \geq u$ and assume that $\mathsf{I} \geq \mathrm{l}$ holds up to v, included. We show that $\mathsf{I} \geq \mathrm{l}$ holds at $v+1$ as well, by case discussion at v for the predicates F and L. More precisely, we have the following cases.

1. $\mathrm{NowOn}(\neg\mathsf{F})$, that is F is false at $v+1$. Then, by Axiom **filling_behavior**, $\mathsf{I} \geq \mathrm{t} > \mathrm{l}$ at $v+1$.

2. $\mathrm{NowOn}(\mathsf{F})$, that is F is true at $v+1$. Then, we distinguish two more cases.

(a) F at v. Therefore, we have immediately $\mathsf{l} \geq \mathrm{l}$ at $v+1$ by Theorem 15.

(b) $\neg\mathsf{F}$ at v. Therefore, by Axiom **filling_behavior**, $\mathsf{l} \geq \mathrm{t}$ at v. Then, by Theorem 17, $\mathsf{l} \geq \mathrm{l}$ at $v+1$.

□

It can be noticed that the proof is remarkably simple. In particular, thanks to the sampling invariance conditions it has been possible to abstract away from all the details on the continuous variability of the l item, while being guaranteed that nothing "unpleasant" happens between any two discrete-time instants. Thanks to the simplicity, the proof can also be completely automated (discuss examples in TRIO/SPIN and/or TeMP). For a comparison, the reader may consider [FR04], where a similar example of a controlled reservoir was verified in continuous-time TRIO exploiting a compositional framework. The system description in [FR04] was less abstract than the present one, and the proof was consequently substantially more complicated even after the simplifications permitted by the compositional approach. In particular, it was not possible to automate the proof process, which could only be checked with the aid of PVS.

Finally, notice that the property, proved in the discrete-time setting, immediately translates through sampling invariance to the theorem **leveLinvariance**.

5.6 Behavior Constraint

Let us now identify the set Ξ of conditions that appear in the sampling invariant formulas above, in order to determine how the derived behavior constraints $\chi^{\phi}_{\bullet}(15-18)$ impacts on the validity of the proved property for general systems.

First of all, there are the conditions predicating about the items F and L, for all possible combination of positive and negated terms. If we think of how the system really operates, it is clear that the filling action F is set by the controller, so it is completely under control. Therefore, we could strengthen the control action so that its value is held over intervals of length δ, which would subsume the condition $\chi^{\phi}_{\bullet}$ for the item F.

The leaking L can, in principle, be highly irregular and change in less than δ time units. However, we can reason informally about the worst-case condition, that is assuming that L is always true. Since in this case the property is verified, as it trivially satisfies the condition $\chi^{\phi}_{\bullet}$, in any other possible behavior where L varies fast, the level will always be *greater than* the value reached in the worst-case scenario. As a consequence, the invariance property will hold for these behaviors as well.

Let us finally consider about the level. The first one is $\mathsf{l} \geq \mathrm{l}$. The behavior constraint $\chi^{\phi}_{\bullet}$ for dense behaviors requires that this formula holds its truth value over intervals which are at least δ time units wide. This is subsumed by the specification *a posteriori*. In fact, if we assume $\chi^{\phi}_{\bullet}$, then Theorem **leveLinvariance** holds, which obviously subsumes the regularity condition, as $\mathsf{l} \geq \mathrm{l}$ is shown to be always true.

Finally, we have the condition $\mathsf{l} \geq \mathrm{t}$. For sake of contradiction, let us assume that $\chi^{\phi}_{\bullet}$ is violated for this condition, that is there exist instants of time in which l goes below t and then back above it in less than δ. For our current verification goal, this behavior is undesirable only as long as, during these δ time units, the level goes below the value l. A simple worst-case reasoning shows however that this cannot happen, since in δ time units l cannot drop by more than $\mathrm{r_l}\delta$, but $\mathrm{r_l}\delta < \mathrm{t} - \mathrm{l}$ by hypothesis.

All in all, for any possible behavior in $b \in [\![\phi]\!]_{\mathbb{R}}$, we have only two cases to consider. Either $\chi^{\phi}_{\bullet}$ holds for b; then Theorem **leveLinvariance** holds by sampling invariance. Or $\chi^{\phi}_{\bullet}$ does not hold for b, but then it is immediate to notice that Theorem **leveLinvariance** holds nonetheless, trivially.

6 Related Works

6.1 Temporal Logics

A nice and readable summary about expressiveness, decidability and complexity results for real-time temporal logics is given by Henzinger [Hen98]. In the remainder, we report on some (more or less) recent result about expressiveness for temporal logic that is not dealt with extensively in [Hen98].

6.1.1 Nesting Operators in Discrete-Time Linear Temporal Logic

The problem of the expressiveness induced by the nesting depth of operators in LTL (with the "traditional" interpretation over discrete time) has been thoroughly studied by Wilke et al. in various works; see [Wil99] for a survey of related results.

In particular, Etessami and Wilke [EW96] have shown that the hierarchy of LTL formulas defined by the maximum depth of nested *until* operators (independently of the number of *next* operators) is expressively strict, and the same holds for LTL formulas where one evaluates the depth of nested *until* and *since* operators. The proof is constructive and based on game-theoretic techniques. More exactly, it is shown that, for all $k \geq 2$ the property denoted as STAIR_k, indicating the strings over the ternary alphabet $\Sigma = \{a, b, c\}$ such that there is a substring where a occurs k times but no b occurs, is expressible with LTL formulas with $k-1$ nested *until* operators, but it is not expressible with fewer than $k-1$ nestings. We note that the given LTL formula for STAIR_k uses $k-1$ nested *next* operators as well. A similar result holds if we allow the past *since* operator as well.

Thérien and Wilke further characterize the so-called "*until-since* hierarchy" of LTL in [TW04] by algebraic means. Without describing the details which are quite convoluted and technical, let us just mention that any LTL property is expressible with a formula with nesting depth of k if and only if the (syntactic) semigroup associated with the property (considered as a formal language) belongs to a certain (decidable) class of finite semigroups.

Kučera and Strejček [KS05] tackle similar problems as [EW96], but from a more "practical" (and less algebraic) perspective. [KS05] can be seen as a nice generalization of previously known results about the *stuttering invariance* of LTL. Let us start by noting that we consider future-only LTL, which basically consists of all formulas expressible with the two modalities *next* (X) and *until* (U). Note that we consider a *weak until*, i.e., one which does include the current instant; therefore *next* is not reducible to it. For this definition of LTL, we define a *strict syntactic hierarchy* of formulas, according to the maximum number of nested *until* and *next* operators in a formula: LTL($\mathsf{U}^m, \mathsf{X}^n$) denotes the set of all LTL formulas where the nesting depth of *until* is at most m, and the nesting depth of *next* is at most n. If one of the two superscript is omitted, we place no restrictions on the nesting depth of that operator: for instance LTL(U^m, X) denotes the set of all formulas where there are at most m nested *until*s, and any number of nested *next*s. The natural question, answered affirmatively in [KS05], is whether the above hierarchy is also *semantically strict*. The answer constitutes an extension of the classical results by Lamport [Lam83], and Peled and Wilke [PW97] about stutter-free languages.

In order to state the main result of the paper, we have to briefly introduce the notion of (m, n)-stutter equivalence. Let $\Sigma^\omega \ni \alpha = \alpha_0\alpha_1\alpha_2\cdots$ an ω-word of alphabet Σ. For $m, n \geq 1$, a finite subword $\beta = \alpha_i\alpha_{i+1}\cdots\alpha_{i+(j-1)}$ of α is (m, n)-redundant whenever the subword $\alpha_{i+j}\alpha_{i+j+1}\cdots\alpha_{i+j+(mj-(m-n-1))-1}$ is a prefix of β^ω. Informally, the subword β is redundant if it is repeated "basically" $m+1$ times, except for a suffix of length $m-1-n$ which can be ignored. We naturally introduce a partial order relation between words according to whether one can be obtained from another by removing some (even infinitely many) non-overlapping (m, n)-redundant subwords. The partial order induces a least equivalence among words, which we call (m, n)-stutter equivalence.

Let us now state the main results of the paper.

- every LTL($\mathsf{U}^m, \mathsf{X}^n$) language is closed under (m, n)-stutter equivalence;
- a language is definable in LTL(U, X) iff it is (m, n)-stutter closed for some $m, n \geq 1$;
- the following results are corollaries of the above main facts, but we report them explicitly for their significance:
  - for all $n \geq 1$, the LTL($\mathsf{U}^0, \mathsf{X}^n$) formula $\overbrace{\mathsf{X}\mathsf{X}\cdots\mathsf{X}}^{n \text{ times}} p$ cannot be expressed in LTL($\mathsf{U}, \mathsf{X}^{n-1}$);
  - for all $m \geq 1$, the LTL($\mathsf{U}^m, \mathsf{X}^0$) formula $\mathsf{F}(p_1 \wedge \mathsf{F}(p_2 \wedge \cdots \wedge \mathsf{F}(p_{m-1} \wedge \mathsf{F}p_m)\cdots))$ cannot be expressed in LTL($\mathsf{U}^{m-1}, \mathsf{X}$);[17]
  - for all $m, n \geq 1$, the classes LTL($\mathsf{U}^{m-1}, \mathsf{X}^n$) and LTL($\mathsf{U}^m, \mathsf{X}^{n-1}$) have incomparable expressive power.

[17] Assuming the usual definition for the *eventually* operator: $\mathsf{F}p \equiv \top\mathsf{U}p$

6.1.2 Decidability of MTL with Finite-Length Point-Based Semantics

Ouaknine and Worrel investigate in [OW05] some questions on the decidability of the temporal logic MTL with a point-based semantics. It is known that MTL is undecidable for interval-based semantics, as it has been discussed in [AH92b, AH93, Hen98]. In [AFH96] it has been demonstrated that the key property of MTL that renders it undecidable is its possibility of expressing punctuality, that is *exact* time distances between events. Due to the denseness of the time domain, undecidability carries over to finite interval-based semantics (see also [DP06]).

Somewhat surprisingly, things change when considering a point-based (timed word) semantics. First of all, notice that if we allow past operators in the language (i.e. we consider $\mathrm{MTL}^{\mathrm{P}}$), undecidability stays (supposedly, with finite domains as well) [Hen91]. As pointed out in [OW05, Footnote 3], this is a consequence of the undecidability proof in [AH93, Hen91], which was carried out in a monadic second-order theory of event sequences, which subsumes both future and past operators. However, the authors show that MTL over finite timed words is instead decidable.

The proof goes through the use of timed alternating automata (TAA, also introduced in [LW05]), that are an extension of the traditional idea of alternating automata to the timed case. More precisely, let us consider TAA with just one clock; this restriction is necessary in order to attain decidability, since the emptiness problem is undecidable for general TAA, but decidable for single-clock ones. Notice that, since the class TAA is closed under all Boolean operations (and complementation, in particular), the universality problem is also decidable for one-clock TAA. Then, the authors show how to construct a one-clock TAA that accepts the same language as that of any given MTL formula. By the way, notice that the construction exploits strict semantics [Hen98] for the *until* operator of MTL, but it is easy to pass to the weak semantics while keeping the same results. Finally, notice that, again due to closure under complementation of MTL formulas, the model-checking problem is equivalent to satisfiability.

[OW05] also discusses additional results:

- the complexity of deciding MTL formulas over finite timed words is non-primitive recursive (this means that, while being recursive, the complexity measure grows faster than any primitive recursive function; it also implies that it is nonelementary — as elementary languages are a strict subset of primitive recursive ones);
- the *safety* fragment of MTL (defined as the fragment of formulas where all intervals appearing in bounded until operators are bounded) is decidable as well on infinite timed words;
- the problem of decidability for full MTL over infinite timed words is settled in [OW06] by the same authors; in this paper, they observe that

their proofs could be extended to handle this case as well, if they endowed TAA with *weak parity acceptance conditions*. While the same with full co-Büchi acceptance conditions would have an undecidable language emptiness problem, parity acceptance conditions are weaker than co-Büchi's.

6.1.3 Undecidability of MTL with Infinite-Length Point-Based Semantics

Ouaknine and Worrel investigate in [OW06] the "missing" result about the decidability of MTL, namely whether satisfiability is decidable for MTL over an infinite-length point-based semantics. The answer is negative, so the decidability results over finite timed words [OW05] do not carry over to the infinite case. More precisely, it is shown that both satisfiability and model-checking (against timed automata) are undecidable for MTL over infinite-length timed words, even if we limit ourselves to the syntactic fragment with constrained *next*[18], *eventually*, and *always* operators. As a side result, combining these results with the open problems about one-clock alternating timed automata discussed in [OW05], one also gets that the emptiness problem for one-clock timed alternating automata with weak parity acceptance conditions is undecidable; *a fortiori*, this implies the undecidability of the same problem for the more expressive one-clock timed alternating automata with Büchi acceptance conditions.

The main results of the paper are drawn by showing how to translate satisfiability questions for MTL (for both infinite and finite point-based semantics) formulas into recurrent-state[19] and halting problems for a particular class of state machines known as *insertions channel machines with emptiness testing*, or ICMETs for short. ICMETs are queue machines with finite-state control and with insertions errors, that is where channels (where data is accessed with a FIFO policy as in a queue) may nondeterministically insert or remove symbols. Channel machines without insertion errors are as expressive as Turing machines; faulty channel machines are instead less powerful, to the extent that their halting and recurrent-state problems are decidable with polynomial-time complexity. By endowing faulty channel machines with emptiness testing (i.e., the ability to test whether a channel is empty), we get a class of automata of intermediate power: their halting problem is decidable (albeit with non-primitive recursive complexity), whereas their recurrent-state problem is still undecidable. Therefore, the aforementioned correspondence between MTL and ICMETs immediately translates to the stated (un)decidability results.

6.1.4 $\mathrm{MITL}^{\mathrm{P}}_{[a,b]}$ with Finite-Length Interval-Based Semantics

Maler, Nickovic and Pnueli consider in [MNP05] the language $\mathrm{MITL}^{\mathrm{P}}_{[a,b]}$, which is a bounded future-and-past version of the temporal logic MITL [AFH96]. More precisely, it is MITL enhanced with past modalities (i.e., the bounded

[18] For timed words, this means referencing to the next timestamped symbol in the word.
[19] That is, determining whether a state is visited infinitely often in a computation.

since operator), with the restriction to temporal intervals of the form $[a, b]$, with $0 \leq a < b$ such that $a, b \in \mathbb{N}$. The language $\text{MITL}^{\text{P}}_{[a,b]}$ is interpreted over finite-length Boolean signals (also called state sequences), that is behaviors b that are (total) functions from $[0, r)$ to $\mathbb{B}^n$ for some $r \in \mathbb{N}$, with non-Zeno requirements. Notice that the restriction to natural numbers is basically the same as the common restriction to rationals, after a proper scaling. Moreover, since we consider finite-length signals, the restriction to finite intervals in MITL formulas is also irrelevant. Thus, we are basically dealing with the same logic as in [AFH96], except that we enhance it with past modalities and restrict it to finite-length state sequences.

About $\text{MITL}^{\text{P}}_{[a,b]}$ with the finite-length interval-based semantics, the authors prove the following results:

- past-$\text{MITL}^{\text{P}}_{[a,b]}$ is deterministic, that is for every past-$\text{MITL}^{\text{P}}_{[a,b]}$ formula π one can construct a *deterministic* timed automaton accepting exactly the behaviors satisfying π;
- future-$\text{MITL}^{\text{P}}_{[a,b]}$ is nondeterministic, that is there exist future-$\text{MITL}^{\text{P}}_{[a,b]}$ formulas ϕ that are accepted by some nondeterministic timed automaton but by no deterministic timed automaton; notice that this implies *a fortiori* the same result for MITL over infinite *interval* sequences.

The reason for this asymmetry turns out to be related to the fact that past formulas have a syntactic idiosyncrasy that guarantees that "fast" changes (i.e., changes whose duration is less than the size of the smallest interval appearing in formulas) can be forgotten as they do not influence the truth of past formulas. This is also due to the fact that we avoid punctual intervals (this is MITL's fundamental feature), otherwise the asymmetry between past and future would not arise.

Finally, the authors point out that determinizability may be obtained by "enforcing some minimal delay of sub-formulas that imply something toward the future"; this requirement is captured by our notion of behavior constraint for formulas in normal form [FR05], or, equivalently, by the inertial bi-bounded delay operator of Maler and Pnueli [MP95].

6.1.5 MITL with Infinite-Length Interval-Based Semantics

Maler, Nickovic and Pnueli in [MNP06] consider once more the language MITL, and propose a translation of MITL formulas to timed automata alternative to the first one proposed by Alur et al. in [AFH96]. Their alternative translation is simpler and is therefore much more amenable to implementation than the original one.

In accordance with [AFH96], the authors choose an infinite-length interval-based semantics of signals. With respect to the semantics assumptions, there are only two departures from what chosen in [AFH96]. First, they disallow punctual behaviors in signals; more precisely, they consider behaviors over $\mathbb{R}_{\geq 0}$ that are constant over left-closed right-open intervals. Second, they consider a

variation of the (bounded) until operator such that whenever $p \mathsf{U} q$ is true, then p must hold as well when q holds in the future, not just before. The authors claim that both assumptions simplify the exposition without appreciable sacrifice of generality.

Correspondingly, the authors show how to build a *timed signal transducer* for any MITL formula. It is a nondeterministic — as future MITL operators are in general nondeterministic [MNP05] — timed automaton that takes as input a signal, representing a behavior of primitive items, and outputs a Boolean signal corresponding to the truth value of the formula over time for the given input. The construction is based on two basic ideas. The first is *modularity*: the structure of the automaton mirrors closely the structure of the formula it models. Moreover, the construction builds on similar work for untimed automata [KP05]. The second in an observation, also made in [AFH96], that Boolean signals representing the truth over time of any MITL formula have the property of holding their truth values for at least δ time units, where δ is the size of the smallest interval appearing in the formula. This property is crucial in ensuring that the testing the truth of signals that range over a continuous domain can be done with finitely many clock, even if, *a priori*, a signal could change its value an infinite number of times in any finite interval.

6.1.6 MTL over Finite-Length Point-Based and Interval-Based Semantics

D'Souza and Prabhakar in [DP06] compare the expressiveness of MTL in the point-based and interval-based (or, as they call it, "continuous") semantics over finite-length interpretations. It is known that MTL is undecidable over interval-based semantics [AH93, AFH96], whereas it is decidable over finite-length point-based (i.e., finite timed words) semantics [OW05]. The authors exploit this difference to show that, over finite-length interpretations, interval-based MTL is more expressive than point-based MTL. Let us add some detail about how they get to the result.

First of all, they introduce a point-based semantics (they call it *pointwise* MTL, and denote it as $\mathrm{MTL}^{\mathrm{pw}}$), and a *continuous* semantics (denoted as $\mathrm{MTL}^{\mathrm{c}}$) which is different from the traditional interval-based semantics. In fact, both semantics interpret MTL formulas over finite-length timed words, so that the two semantics can be put on a common ground and compared. The difference is that, while in pointwise semantics each formula can predicate only about instants of time that correspond to events in the word, in the continuous semantics formulas assert at all instants of time, including in between any two timestamped events.

It is fairly easy to show that $\mathrm{MTL}^{\mathrm{c}}$ is at least as expressive as $\mathrm{MTL}^{\mathrm{pw}}$. In order to show that the inclusion is strict, the authors consider the undecidability proof for $\mathrm{MTL}^{\mathrm{c}}$, which relies on encoding deterministic 2-counter machines. Then, the separation proof proceeds by contradiction: if one assumes that $\mathrm{MTL}^{\mathrm{pw}}$ is as expressive as $\mathrm{MTL}^{\mathrm{c}}$, then it must be possible to perform the same encoding of 2-counter machines in $\mathrm{MTL}^{\mathrm{pw}}$. But then, the satisfiability

problem for MTL^{pw} would undecidable, a contradiction of a previously known result [OW05].

Finally, the authors briefly comment on how the same technique could be used to differentiate the expressiveness of other formalisms; for instance they show that the expressive powers of one-clock alternating timed automata and k-clock nondeterministic timed automata are incomparable.

6.1.7 TPTL, MTL, and MTL^{P} over Infinite-Length Point-Based and Interval-Based Semantics

Bouyer et al. [BCM05] analyze the expressiveness of the two well-known real-time temporal logics TPTL [AH94] and MTL [Koy90, AH93]. More precisely, they carefully analyze a long-held conjecture [AH92b, AH93, Hen98] about TPTL being more expressive than MTL over dense time.

The main contribution paper is an in-depth proof of the conjecture for both the point-based and the interval-based semantics. However, it is shown that the formula $\Box(\mathsf{p} \Rightarrow x.\Diamond(\mathsf{q} \wedge \Diamond(\mathsf{r} \wedge x \leq 5)))$ proposed in [AH93] as a witness of the conjecture is actually not expressible in MTL *only* in the point-based semantics. The authors, however, provide a different witness (namely, the formula $\Box(\mathsf{p} \Rightarrow x.\Diamond(\mathsf{q} \wedge x \leq 1 \wedge \Box(x \leq 1 \Rightarrow \neg\mathsf{r})))$) which separate the expressiveness classes in the interval-based semantics as well. The result is very technical and quite convoluted (as most expressiveness results are), so we do not report here the low-level details, for simplicity.

Let us remark, however, that both families of models that separate TPTL and MTL in the point-based or interval-based semantics are (simply) characterizable using *past* operator of MTL's. Therefore, a corollary of the expressiveness results for TPTL and MTL is that MTL^{P} (i.e., MTL with past operators) is strictly more expressive than future-only MTL in both the point-based and the interval-based semantics.

Finally, it is shown that the existential fragments of TPTL and MTL — defined as the fragments where the only allowed modality is $\Diamond$ — are fully decidable, and the two language fragments are *equally* expressive.

6.1.8 MTL^{P} over Infinite- and Finite-Length Point- and Interval-Based Semantics

Prabhakar and D'Souza present several results about the expressiveness of MTL, and syntactic and semantic variants thereof, in [PD06]. Several of the results use similar techniques as those in [BCM05], while generalizing and extending them to handle other cases.

The authors consider four basic semantics for the temporal logic languages they analyze: point-based semantics (called *pointwise* in the paper), a variant of interval-based semantics (called *continuous* semantics; basically it consists of an interval-based semantics where all items are constrained to behave as instantaneous *events*. This is the same semantics as in [DP06].), and both infinite- and finite-length variants of the two semantics. Concerning syntax,

they consider three variants of the MTL language: one is "regular" MTL (i.e., with the constrained *until* as only modality); then we have MTL_S, which is MTL enriched with the *unconstrained since* operator for the past; and finally we have MTL_{S_I}, which is MTL with both constrained *until* and constrained *since* operators.

Before succinctly summarizing the results of the paper, let us briefly present a preliminary result which is used in the paper to extend known results about the expressiveness of MTL over infinite behaviors (and in particular, those in [BCM05]) to finite-length behaviors as well. The result, which holds over both the pointwise and the continuous semantics, is a sort of *counter-freeness* property of MTL, which can be regarded as an analogue of the counter-freeness characterization of LTL over discrete time domains [MP72] (see also [Wil99] for a brief history of the subject). Informally, the result can be stated as follows. Given an (infinite) sequence of finite timed words $\sigma_0, \sigma_1, \sigma_2, \ldots$, which are *periodic* (i.e., one can be obtained from a previous one by "pumping" a fixed finite subword), we say that the sequence *ultimately satisfies* an MTL formula ϕ if from some index $k \geq 0$ all words in the sequence satisfy ϕ. If, from some index $k \geq 0$, all words in the sequence do not satisfy ϕ, we say that the sequence *ultimately does not satisfy* ϕ. The counter-freeness property says that, given an MTL formula ϕ, all periodic sequences of finite timed words either ultimately satisfy or ultimately do not satisfy ϕ. This characterization is useful in extending expressiveness results for MTL to finite words.

Finally, here it is a short summary of the expressiveness results of the paper.

- MTL in the continuous semantics is strictly more expressive then MTL in the pointwise semantics, for both finite and infinite behaviors; here, the proof for infinite behaviors of [BCM05] is extended to the finite case;
- in the continuous semantics, MTL is strictly contained in MTL_S, for both finite and infinite behaviors; again, the proof for infinite behaviors of [BCM05] is extended here to the finite case (in accordance with the results in [DP06]);
- the language MTL in the continuous semantics is strictly less expressive than the language MTL_S, over both finite and infinite behaviors;
- the language MTL_S is strictly less expressive than MTL_{S_I} in the pointwise semantics, over both finite and infinite behaviors; note that whether the same inclusion is strict in the continuous semantics is still unknown.

6.1.9 Expressive Completeness of Future Linear Temporal Logic

Hirshfeld and Rabinovich consider in [HR03] the problem of finding a (untimed) temporal logic which is expressively complete for the future fragment of the monadic logic of order (MLO).

MLO is traditionally considered the standard setting in which ordering properties about linear sequences of events can be expressed. Similarly, Linear Temporal Logic is traditionally considered a language suitable to express the same

properties as with MLO, but in a much more natural and intuitive way. In other words, the use of modalities in place of first-order logic with ordering predicates can be considered as a way of "syntactic-sugaring" the underlying MLO. The sugaring is desirable as long as the modal logic language is expressively complete with respect to MLO. For LTL the expressive completeness has been first proved by Kamp [Kam68]. Kamp's result holds for LTL with the *until* and *since* modalities, for MLO interpreted over Dedekind-complete structures[20]; in particular the standard models of natural and nonnegative real numbers — with respect to standard ordering — are comprised in Kamp's result.

[HR03] considers the derived problem of finding a temporal logic, using only future operators, which is expressively complete for the *future fragment* of MLO, which is characterized in a precise way. This apparently simple restriction yields nonetheless surprising results. First of all, the use of the sole *until* operator does not bring expressive completeness over the reals (whereas it is well-known that it does so over the naturals [GPSS80]). Even more surprisingly, [HR03] shows that no temporal logic with a finite number of future modal operators can be expressively complete for future properties. Note, however, that this limitation holds only for general behaviors over the reals (and rationals as well). If we restrict our analysis to finitely-variable (i.e., non-Zeno) behaviors only — as it is routinely done — then the expected result (that *until* is expressively complete) holds.

6.2 Timed Automata

6.2.1 Two-Way Timed Automata with Finite-Length Point-Based Semantics

Alur and Henzinger consider in [AH92a] *two-way timed automata*, that is timed automata that can go back and forth on a timed word. They adopt an point-based (i.e., timed word) finite-length semantics. The basic nondeterministic version of these automata is denoted as 2NTA. Their deterministic subset is denoted by 2DTA; notice that determinism ensures that there is a unique run for any timed word, but a 2-way deterministic timed automaton may become nondeterministic when rendered as a (1-way) traditional timed automaton. Let us also introduce the notion of bounded timed automata: the class 2NTA_k is the subclass of two-way (nondeterministic) timed automata such that in every run over any timed word the automaton visits any symbol in the word at most $2k+1$ times; the class 2DTA_k is its deterministic counterpart. Notice that standard nondeterministic and deterministic timed automata (usually denoted by NTA and DTA, respectively) correspond to the classes 2NTA_0 and 2DTA_0, respectively.

The authors draw the following results:

- the automata in each class 2DTA_k give a strictly less expressive formalism than those in the class 2DTA_{k+1};

[20] That is such that every non empty bounded subset has a lowest upper bound.

- conversely, increasing the number of nondeterministic reversals does not add expressiveness to that achieved by 2NTA_0 (i.e., NTA);
- since deterministic (two-way) timed automata are closed under all Boolean operations, while NTA are not, and it is easily seen that one.way nondeterminism can simulate any finite number of deterministic reversals, the class of languages defined by 2DTA_k for any $k \in \mathbb{N}$ is strictly contained in the class of languages defined by NTA;
- for every MITL formula ϕ, there exists an automaton in 2DTA_1 that accepts precisely the behaviors that satisfy ϕ; the converse is not true, as there exist MITL formulas that require nondeterminism (see also [MNP05]);
- let us consider MITL^{P}, that is MITL enriched with past operators. As the number of alternations in the nesting of subformulas between past and future modalities in a MITL^{P} formula increases, the number of reversals required by an automaton in 2DTA_k — accepting the same language as the formula — increases correspondingly. A byproduct of this result is that MITL remains decidable even if enriched with past operators;
- let us finally consider the language $\text{MITL}_{=}$, the extension of MITL that allows singular (i.e., pointwise) intervals in temporal operators; recall that in [AFH96] it was shown that this introduces undecidability over infinite-time interval-based interpretations, but also see [OW05] for related results with different interpretations. The authors show that the models of any $\text{MITL}_{=}$ formula can be recognized by some automaton in 2DTA, but not by any finite number of reversals (i.e., an unbounded number is required in the worst case);
- the authors also *conjecture* that unbounded deterministic two-wayness is not powerful enough to model nondeterminism, even if the latter is without reversals; they also propose a language that they believe cannot be accepted by any 2DTA, but is recognized by a NTA.

6.2.2 Event-Clock Automata with Finite-Length Point-Based Semantics

Alur, Fix and Henzinger introduce in [AFH99] three classes of related variations of timed automata: event-recording automata, event-predicting automata, and event-clock automata. The semantics they consider is over point-based sequences of finite length, that is finite timed words.

Event-recording automata are automata endowed with clock that record the time of the last occurrence of any event in the alphabet. *Event-predicting* automata have instead clock that "predict" the time of the occurrence of an event. Finally, *event-clock* automata have both recording and predicting clocks. We let ERA, EPA, and ECA denote the classes of even-recording, event-predicting, and event-clock automata, respectively.

Event-clock automata have the nice properties of being determinizable, and so are the subclasses of event-recording and event-predicting automata. Moreover, their emptiness problem is decidable, through a region construction that draws from the corresponding one for timed automata [AD94], and the languages they accept are closed under all Boolean operations. Also notice that event-clock automata are instead not closed under renaming or hiding of input symbols (contrarily to (nondeterministic) timed automata).

The authors draw the following results about the expressiveness of the various classes of automata they have introduced:

- the classes of languages recognized by automata in ERA and EPA are incomparable;
- the possibility of having both predicting and recording clocks in the same automaton (as in ECA) gives stronger expressive power than that given by union of ERA and EPA;
- ECA are strictly less expressive than nondeterministic timed automata NTA (obviously, as the former are determinizable, while the latter are not):
- ERA can be fully translated to deterministic timed automata DTA, while the converse is in general not true;
- EPA and DTA yield language classes that are incomparable;
- ECA and DTA also yield language classes that are incomparable.

6.2.3 Alternating Timed Automata with Finite-Length Point-Based Semantics

Lasota and Walukiewicz introduce in [LW05] the notion of alternating timed automata (denoted as ATA). An ATA is a classic alternating automaton [CKS81, Var97] enriched with clocks, just like timed automata are an enrichment of classic automata. Note that ATA were also independently introduced by Ouaknine and Worrel [OW05]. The authors give a game-theoretic semantics for ATA, based on finite-length timed words (i.e., a point-based semantics).

Languages accepted by ATA are easily shown to be closed under all Boolean operations. Moreover, it is also simple to show that the emptiness problem is undecidable for ATA with more than one clock: the universality problem for timed automata with the same number of clocks — well-known to be undecidable [AD94] — can be easily encoded as an emptiness problem for ATA. Therefore, the paper focuses on one-clock ATA. For this class of automata, the following results are drawn.

- emptiness (and thus, as a simple corollary deriving from closure under Boolean operations, universality and language containment as well) is decidable for one-clock ATA. The proof relies on a region construction that builds a finite bisimilar transition system;

- the complexity of the emptiness problem is non-primitive recursive;
- ATA enriched with ϵ-moves become undecidable even with one clock: this is shown by reduction from the reachability problem for perfect channel systems (whereas without ϵ-moves one can only encode lossy channel systems);
- the authors also briefly hint at the fact that universality for one-clock nondeterministic timed automata over infinite words (and thus for ATA over infinite words as well) is undecidable; this is given without proof, but it is discussed in [ADOW05].

Finally, the authors also plan to find different syntax that, while allowing the same expressiveness as ATA, will give more treatable complexities for the emptiness problem; one might speculate that this could pass through a logical characterization of the same language class.

6.2.4 1-Clock Timed Automata with Finite and Infinite Point-Based Semantics

Abdulla et al. [ADOW05] study the language inclusion problem for timed automata with one clock. More precisely, they consider a point-based (i.e., timed word) semantics and study the problem of deciding, given a 1-clock TA A and another TA B (with no restrictions on B), whether every word accepted by B is also accepted by A. Previously, Alur and Dill [AD94] showed the problem undecidable for A with two or more clocks, and decidable for A without clocks.

The results drawn in the paper can be summarized as follows; they all exploit reductions from reachability problems in channel machines [AJ96, CFP96, Sch02].

- over finite-length words, the one-clock language inclusion problem is decidable with non-primitive recursive complexity. This was already shown by the same authors in [OW04];
- if we allow ϵ-transitions, the problem is undecidable even for one-clock automata over finite words;
- the problem is also undecidable for one-clock timed automata over infinite words with Büchi acceptance conditions.

Note that for timed automata, contrarily to untimed "classic" automata, Büchi conditions are stronger than the so-called *weak parity acceptance conditions*. Only the latter are needed in order to encode MTL formulas with timed automata with one clock. Still, the satisfiability problem for MTL is not be decidable over infinite words [OW06].

6.3 Other Formalisms

6.3.1 A Decidable Monadic Second-Order Logic

Wilke considers in [Wil94] a decidable monadic second-order logic, which is suitable to express properties of timed state sequences. The starting point is the *monadic logic of distance*, denoted by $\mathcal{L}_{\mathrm{d}}$; this is a monadic second-order language which allows one to predicate about distances between instants in timed words. Therefore, notice that the work assumes a point-based infinite-length semantics (i.e., timed ω-words). This language is undecidable, as the same results of [AH93] carry over for $\mathcal{L}_{\mathrm{d}}$.

Thus, the author introduces a fragment of $\mathcal{L}_{\mathrm{d}}$ where the use of expressions about distance is restricted to relative distances only. The resulting fragment, called the *monadic logic of relative distance*, is denoted by $\mathcal{L}_{\overleftrightarrow{\mathrm{d}}}$. $\mathcal{L}_{\overleftrightarrow{\mathrm{d}}}$ is shown to define exactly the same class of timed languages as timed automata. Therefore, it is a logical characterization of timed automata, and its satisfiability problem is decidable (by building the corresponding timed automaton and checking for emptiness).

Being as expressive as timed automata, $\mathcal{L}_{\overleftrightarrow{\mathrm{d}}}$ is in general not closed under complement. However, if we restrict the interpretation of formulas in the language to timed words with *bounded variability* (i.e., such that there is an upper bound on the number of events that can occur in any finite-length interval), then the resulting class of languages is closed under complement. Clearly, the resulting theory is also fully decidable.

Finally, the author embeds some known temporal logics in the language $\mathcal{L}_{\overleftrightarrow{\mathrm{d}}}$, thus showing their effective decidability. The chosen logics are TL_{Γ} [MP93], $\mathrm{MITL}^{\mathrm{P}}$ (i.e., MITL with past operators) and an extension thereof called $\mathrm{EMITL}^{\mathrm{P}}$ (which extends $\mathrm{MITL}^{\mathrm{P}}$ with *automata operators* [WVS83]).

6.3.2 The Regular Real-Time Languages

Henzinger et al. [HRS98] provide a classification of (mostly decidable) real-time formalisms, interpreted over timed state sequences (i.e., for interval-based semantics). The paper summarizes previous results, and adds new contributions to the picture.

Basically, the paper identifies three levels of expressiveness. Each level is represented by one (or more) temporal logic languages, a monadic logic theory, and variants of timed automata. Let us briefly consider them.

- The lowest expressiveness level is labeled $\mathbb{R}$-*timed counter-free ω-regular*, and represents a fully decidable class of languages.

  The corresponding monadic theory is called $\mathrm{MinMaxML}_1$; in a nutshell, it is a first-order sequential calculus over timed sequences, endowed with two basic quantifiers $\min, \max$ (with the intuitive meanings) for quantification over the time sort (as a side remark, notice that the interpretation is nonstandard, encompassing both undefined values and infinitesimal values to handle open and closed intervals in a uniform manner).

The corresponding temporal logic languages are MITL [AFH96] and SCL [RS97] (named EventClockTL in [HRS98]).

There is no corresponding automata class, as this class of languages cannot perform "modulo-counting" (hence the label *counter-free*), whereas automata can (by exploiting states).

Concerning the complexity of the decidability problem, it is nonelementary for MinMaxML$_1$, **PSPACE**-complete for SCL, **EXPSPACE**-complete for MITL. More practically, one may say that the two expressively equivalent languages MITL and SCL are orthogonal for what concerns the ease and naturalness of expressing properties of interest.

- Next, we have a class labeled $\mathbb{R}$-*timed* ω-*regular*, which is still fully decidable.

  The corresponding monadic theory is called MinMaxML$_2$; it is a second-order extension of MinMaxML$_1$, where second-order quantification over monadic predicates is allowed. Note that quantification is still restricted to predicates not occurring within the scope of a min or max quantifier.

  At the same level of expressiveness we have two equivalent extensions of both MITL and SCL that achieve *counting* for the logics. The former consists in using automata connectives (like those introduced in [WVS83]); the corresponding logics are labeled E-MITL and E-EventClockTL. The latter consists in introducing second-order quantification over predicates; a restriction (analogous to that of the monadic logic) requires that the quantified predicates do not occur in any of SCL's history or prophecy operators (denoted as $\lhd$ and $\rhd$, respectively). The resulting logics are labeled Q-MITL and Q-EventClockTL.

  The corresponding automata class if that of event-clock timed automata ECA [AFH99]; more precisely, an equally-expressive variation of them, which allows for recursive definition, is considered in [HRS98].

  Concerning the computational complexity of decidability, it is the same as that of the counter-free level of expressiveness.

- If we still increase the expressiveness of formalisms we get the class labeled *projection-closed* $\mathbb{R}$-*timed* ω-*regular*; as the name suggests, it is achieved by introducing *projection* in our languages. The resulting theories are only positively decidable, that is satisfiability is decidable, but validity is not. This obviously implies that the corresponding formalisms are closed under positive Boolean operations, but not closed under complement,

  More precisely, we introduce a monadic theory called P-MinMaxML$_2$; which is obtained by allowing outermost existential second-order quantification (i.e., projection) over monadic predicates occurring within the scope of a min or max quantifier. This monadic theory is as expressive as Wilke's $\mathcal{L}_{\overleftrightarrow{\mathrm{d}}}$ theory [Wil94].

One can introduce the corresponding projection operators for the logic SCL, getting the language P-EventClockTL, and for event-clock automata, getting the automata class P-ECA. Notice that P-ECA correspond to "vanilla" timed automata [AD94], for which it is well known that emptiness is decidable, and universality is undecidable.

- Finally, one may permit full quantification over monadic predicates; the corresponding formalisms would be equivalent in expressive power to Boolean combinations of timed automata, and it would be fully undecidable.

6.3.3 Decidable Metric Temporal Logics over the Real Line

Hirshfeld and Rabinovich in [HR04] (and, previously, in [HR99b, HR99a]) present some results about decidability and expressiveness of modal logics for real-time, comparing them with other well-known contributions in the same area. The followed approach is rather different than those of most — if not all — other similar works, and it is insightful in considering common problems and questions from a different perspective.

The semantic base is the canonical time model of the nonnegative real numbers. Notice that one of the main features of the work is that we do not restrict ourselves to non-Zeno behaviors over such a time model, but consider any possible behavior of Boolean-valued monadic predicates (i.e., time-dependent propositions, with another wording). From a purely mathematical point of view, it is quite natural to introduce a monadic language to express metric properties in these models. Since qualitative properties are naturally expressed with the monadic logic of order (MLO), its straightforward generalization that introduces a unary function $+1$ over the time sort is considered as the "standard" language in which to express quantitative temporal properties. We call this language MLO^+.

Since MLO^+ is undecidable, we seek a "sufficiently expressive" subset of it which is decidable. To do this, [HR04] starts by defining a metric modal logic with two metric modalities $\Diamond_1$, $\overleftarrow{\Diamond}_1$, in addition to the standard *until* and *since*. It is called Quantitative Temporal Logic (QTL). The formula $\Diamond_1 \mathsf{p}$ (respectively, $\overleftarrow{\Diamond}_1 \mathsf{p}$) denotes that p will hold (has held) within the next (previous) time unit. The corresponding syntactic fragment of MLO^+, for which QTL is expressively complete, is denoted as QMLO.

Although QTL may seem not very expressive, [HR04] demonstrates that this is not the case. In particular, it is at least as expressive as all known decidable real-time logics (such as MITL). Indeed, QTL is also decidable. [HR04] shows this with purely logical methods, without resorting to automata theory. In a nutshell, the basic idea is to reduce the satisfiability problem of any QTL formula to the satisfiability of another formula in *timer normal form*, using auxiliary variables. The timer normal form is made of a conjunction of a MLO (and thus non-metric) formula, and a particular QMLO formula *Timer* expressing quantitative constraints. Then, it is shown that *Timer* can be further reduced to a pure MLO formula by observing that, roughly speaking,

its satisfiability is preserved by "stretchings" of the real line. Therefore, since MLO is fully decidable, QTL (and QMLO, being as expressive) is decidable as well.

Still resorting solely to logic-based methods, it is shown that the satisfiability problem for QTL is in **PSPACE**. Just like the other results, this holds both for unrestricted behaviors, and for non-Zeno (a.k.a. finitely-variable) ones.

Next, [HR04] considers possible extensions of QMLO and QTL that are more expressive, but still decidable. In particular, let us mention the introduction of a hierarchy of modal logics, of increasing expressive power, that are all decidable: for each $n > 0$, the logic QTL(n) has a modality capable of expressing the fact that n predicates eventually become true (in the given order) within one time unit. Other enrichments using automata connectives and variants thereof are briefly discussed.

In this vein, the other paper [HR06] proves a conjecture by Pnueli, and a generalization of it, about the expressiveness of "counting modalities" similar to those of the logic QTL(n) mentioned in the previous paragraph. Even more generally, it is shown that for any temporal logic — even with infinitely many modalities — all of whose modalities are expressible in the second-order monadic logic of order (enriched with the unary addition predicate to account for metric properties) with a bounded quantifier depth, there exists some n such that the property that a predicate is true at least n times within the next unit of time is not expressible in the logic. This result is strong, even if we have to consider that several variants may open different perspectives. In particular, one may ask what happens if we allow more arithmetic than just addition by one, or, more generally, what other modalities should be considered as "basic". Finally, notice that the general result is proved for the whole real line, even if it is conjectured that it holds for the nonnegative reals as well.

Finally, [HR04] concludes with a critical discussion and comparison with existing similar works. In particular, it remarks how the two widely used ω-models of timed words and timed interval sequences lack some basic features and their customary use has refrained some basic results about more general (and natural) models from being discovered (especially unrestricted behaviors over the reals). On the other hand, the proliferation of different models and temporal logic languages for quantitative properties is mostly due to the shortcoming of such models, in the opinion of [HR04]. As an example, it is remarked how point-based models fail to satisfy intuitively perfectly reasonable properties such as $\Box \overleftarrow{\Diamond}_1 \mathtt{true}$, which is not satisfied by timed words where some pair of timestamps differ by more than one time unit. For what concerns interval-based sequences, although they represent precisely the finite-variability requirement, they are not unique representations, in that the same behavior is representable with more than one choice of interval coverage. Moreover, as it is also well-known [AFH96], the corresponding logics do not distinguish between a real and a rational model; according to [HR04] this is an indication that the accepted models need reconsideration.

6.3.4 Automata over Continuous Time

Rabinovich in [Rab03] considers the problem of "lifting" the classical concepts of automata theory from discrete to continuous time. Differently than most other works with similar objectives, it tries to do so in a very general and abstract way, starting from general concepts and refining them into more concrete notions. Such an approach allows one to reconsider most *ad hoc* solutions that have been provided elsewhere, and to evaluate their respective merits and shortcomings.

The fundamental entity is the signal, which is a mapping from the time domain (namely, the nonnegative reals) to some finite set Σ. Most of the times only non-Zeno signals will be considered, even if some general and useful results about general signals are also drawn. A machine is then described as a device that implements some signal-to-signal function. The author focuses on functions with the following characteristics (that we recall here informally):

- *strongly retrospective*, that is causal functions;
- *finite memory*, that is such that they distinguish their future behavior among only a finite number of classes of signal histories.

Afterward, it is shown how a non-Zeno signal can be represented by means of ω-strings (pairs of them, more precisely) and time sequences, thus introducing a description of signals which is closer to the traditional view of discrete-time automata. Notice, however, that the resulting model is different than the usual timed trace or timed interval sequences.

Then, it is shown that a function with finite memory is necessarily *speed independent*, that is it is invariant under "stretchings" (technically, order-preserving bijections) of the time axis. Therefore, every machine with finite memory is incapable of dealing with metric constraints. On the positive side, it is shown how for finite-memory strongly retrospective functions one can associate a description of a *state* which allows for a natural representation of such functions by means of finite transducers. Therefore, one can see how some familiar notions that are typically introduced as the basis for the formalization of finite-state machines can instead be derived from basic principles. Notice that this helps the understanding of some intrinsic limitations of such models.

Several other notions about signals and functions on signals are introduced and studied in [Rab03], but we do not discuss them here for brevity.

6.3.5 Finite Automata Extensions for Hybrid Systems

Rabinovich and Trakhtenbrot tackle the problem of extending the basic finite automaton model to deal with the description of hybrid systems in [RT97], and they do so with their typical "top-down" approach, with the profound goal of lifting some classical results about automata (or labeled transition systems), nets (i.e., systems of logic equations), and monadic logic (and its "syntactic sugar" temporal logic), which they dub the "trinity" [Tra95]. Namely, they separately consider two extensions: relativization and continuous time. Notice that we recall here the notions and results of [RT97] in a very informal, and thus

also somewhat imprecise, way: our only goal is to convey an informal meaning, while we refer to [RT97] for technical, unambiguous explanations.

Relativization is a fundamental concept in theoretical computer science. To define it, a notion of product between automata and projection of an automaton with respect to an alphabet are introduced. Then, the relativization of an automaton A with respect to another automaton B (called the "oracle), denoted as A^B, is the projection over the alphabet of A which is not in the alphabet of B of the product automaton of A and B. In other words, A^B is an automaton which "delegates" to B the computation needed for the acceptance of some "aspects" of the input.

Then it is considered which of five issues relativize, that is still hold for finite automata with access to some oracle (in general, infinite-state). The issues are:

- whether *monadic logic* is as expressive as the relativized automata;
- whether there exists a finite set of basic operators such that *nets* built out of this set of operators are as expressive as the relativized automata;
- whether relativized automata can always be made *deterministic*;
- whether relativized acceptor automata are as expressive as relativized transducer automata (that is if *uniformization* is possible);
- whether emptiness is decidable for relativized automata.

It is shown that, while the former two are retained for any oracle, the others may hold or fail depending on the particular choice of oracle.

On the other hand, it is shown that for another extension of finite automata, one that deals with continuous time without any relativization, all of the five issues are answered affirmatively for non-Zeno signals.

Finally, notice that most automata extensions that have been proposed in the literature to deal with hybrid systems, can indeed be seen as a relativization and/or continuous-time extension of classical automata, even if they have not been presented as such in the literature. In particular, timed automata [AD94] are basically a finite automata with access to a "clock" oracle.

6.4 Other Notions of Discretization

Asarin, Maler and Pnueli in [AMP98] tackle the problem of discretizing the behavior of digital circuits in a way that preserves the qualitative behavior of the circuit in a strict sense, that is such that the ordering of events in continuous time must be unchanged in the discretized behavior. In particular, this implies that events occurring at different times, however close, must occur at *distinct* discrete time instant, in the same order. This is a stronger notion than — in particular — those by Henzinger et al. [HMP92], where a strictly monotonic sequence is allowed to be discretized into a weakly monotonic one.

More precisely, the authors are basically considering Boolean signals, that is interval-based behaviors over infinite time (i.e., the time domain is $\mathbb{R}_{\geq 0}$). The

same authors have shown in [MP95] that the subclass signals that are definable by their digital circuits can also be accepted by timed automata.

The results about digitizability are demonstrated by geometric arguments about polyhedra. While for the simpler acyclic circuits digitizability is always possible for some suitable choice of discretization step δ, for general cyclic circuits the distance between state changes can become arbitrarily small, so that discretization is impossible for any choice of step $\delta > 0$. On the contrary, the same cyclic circuits are discretizable according to the weaker notion of discretization introduced in [HMP92].

References

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

[ADOW05] Parosh Aziz Abdulla, Johann Deneux, Joël Ouaknine, and James Worrell. Decidability and complexity results for timed automata via channel machines. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, volume 3580 of *Lecture Notes in Computer Science*, pages 1089–1101. Springer-Verlag, 2005.

[AFH96] Rajeev Alur, Tomás Feder, and Thomas A. Henzinger. The benefits of relaxing punctuality. *Journal of the ACM*, 43(1):116–146, 1996.

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

[AH92a] Rajeev Alur and Thomas A. Henzinger. Back to the future: Towards a theory of timed regular languages. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science (FOCS'92)*, pages 177–186. IEEE Computer Society Press, 1992.

[AH92b] Rajeev Alur and Thomas A. Henzinger. Logics and models of real time: A survey. In J. W. de Bakker, Cornelis Huizing, and Willem P. de Roever, editors, *Proceedings of the Real-Time: Theory in Practice, REX Workshop*, volume 600 of *Lecture Notes in Computer Science*, pages 74–106. Springer-Verlag, 1992.

[AH93] Rajeev Alur and Thomas A. Henzinger. Real-time logics: Complexity and expressiveness. *Information and Computation*, 104(1):35–77, 1993.

[AH94] Rajeev Alur and Thomas A. Henzinger. A really temporal logic. *Journal of the ACM*, 41(1):181–204, 1994.

[AJ96] Parosh Aziz Abdulla and Bengt Jonsson. Undecidable verification problems for programs with unreliable channels. *Information and Computation*, 130(1):71–90, 1996.

[AL94] Martín Abadi and Leslie Lamport. An old-fashioned recipe for real-time. *ACM Transactions on Programming Languages and Systems*, 16(5):1543–1571, 1994.

[AMP98] Eugene Asarin, Oded Maler, and Amir Pnueli. On discretization of delays in timed automata and digital circuits. In Davide Sangiorgi and Robert de Simone, editors, *Proceedings of the 9th International Conference on Concurrency Theory (CONCUR'98)*, volume 1466 of *Lecture Notes in Computer Science*, pages 470–484. Springer-Verlag, 1998.

[BCM05] Patricia Bouyer, Fabrice Chevalier, and Nicolas Markey. On the expressiveness of TPTL and MTL. In R. Ramanujam and Sandeep Sen, editors, *Proceedings of the 25th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'05)*, volume 3821 of *Lecture Notes in Computer Science*, pages 432–443. Springer-Verlag, 2005.

[CFP96] Gérard Cécé, Alain Finkel, and S. Purushothaman Iyer. Unreliable channels are easier to verify than perfect channels. *Information and Computation*, 124(1):20–31, 1996.

[CHR02] Franck Cassez, Thomas A. Henzinger, and Jean-François Raskin. A comparison of control problems for timed and hybrid systems. In Claire Tomlin and Mark R. Greenstreet, editors, *Proceedings of the 5th International Workshop on Hybrid Systems: Computation and Control (HSCC'02)*, volume 2289 of *Lecture Notes in Computer Science*, pages 134–148. Springer-Verlag, 2002.

[CKS81] Ashok K. Chandra, Dexter Kozen, and Larry J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981.

[DP06] Deepak D'Souza and Pavithra Prabhakar. On the expressiveness of MTL in the pointwise and continuous semantics. *Formal Methods Letters (Software Tools for Technology Transfer)*, 2006. To appear.

[EW96] Kousha Etessami and Thomas Wilke. An until hierarchy for temporal logic. In *Proceedings of the 11th Annual IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 108–117. IEEE Computer Society Press, 1996.

[FMM94] Miguel Felder, Dino Mandrioli, and Angelo Morzenti. Proving properties of real-time systems through logical specifications and Petri net models. *IEEE Transactions on Software Engineering*, 20(2):127–141, 1994.

[FMMR06] Carlo A. Furia, Dino Mandrioli, Angelo Morzenti, and Matteo Rossi. Modeling time in computing: A taxonomy and a comparative survey. Technical report, Dipartimento di Elettronica e Informazione, Politecnico di Milano, 2006. Submitted for publication.

[FR04] Carlo A. Furia and Matteo Rossi. A compositional framework for formally verifying modular systems. In *Proceedings of the International Workshop on Test and Analysis of Component Based Systems (TACoS'04)*, volume 116 of *Electronic Notes in Theoretical Computer Science*, pages 185–198. Elsevier, January 2004.

[FR05] Carlo A. Furia and Matteo Rossi. When discrete met continuous: on the integration of discrete- and continuous-time metric temporal logics. Technical Report 2005.44, Dipartimento di Elettronica e Informazione, Politecnico di Milano, October 2005.

[FR06] Carlo A. Furia and Matteo Rossi. Integrating discrete- and continuous-time metric temporal logics through sampling. In Eugene Asarin and Patricia Bouyer, editors, *Proceedings of the 4th International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS'06)*, volume 4202 of *Lecture Notes in Computer Science*, pages 215–229. Springer-Verlag, September 2006.

[GHR94] Dov M. Gabbay, Ian Hodkinson, and Mark Reynolds. *Temporal Logic (vol. 1): mathematical foundations and computational aspects*, volume 28 of *Oxford Logic Guides*. Oxford University Press, 1994.

[GM01] Angelo Gargantini and Angelo Morzenti. Automated deductive requirement analysis of critical systems. *ACM Transactions on Software Engineering and Methodology*, 10(3):255–307, 2001.

[GPSS80] Dov M. Gabbay, Amir Pnueli, Saharon Shelah, and Jonathan Stavi. On the temporal basis of fairness. In *Conference Record of the 7th Annual ACM Symposium on Principles of Programming Languages (POPL'80)*, pages 163–173, 1980.

[Hen91] Thomas A. Henzinger. *The Temporal Specification and Verification of Real-Time Systems*. PhD thesis, Department of Computer Science, Stanford University, 1991. Also Technical Report STAN-CS-91-1380.

[Hen98] Thomas A. Henzinger. It's about time: Real-time logics reviewed. In Davide Sangiorgi and Robert de Simone, editors, *Proceedings of the 9th International Conference on Concurrency Theory (CONCUR'98)*, volume 1466 of *Lecture Notes in Computer Science*, pages 439–454. Springer-Verlag, 1998.

[HMP92] Thomas A. Henzinger, Zohar Manna, and Amir Pnueli. What good are digital clocks? In Werner Kuich, editor, *Proceedings of the 19th International Colloquium on Automata, Languages and Programming (ICALP'92)*, volume 623 of *Lecture Notes in Computer Science*, pages 545–558. Springer-Verlag, 1992.

[HR99a] Yoram Hirshfeld and Alexander Moshe Rabinovich. A framework for decidable metrical logics. In Jirí Wiedermann, Peter van Emde Boas, and Mogens Nielsen, editors, *Proceedings of the 26th International Colloquium on Automata, Languages and Programming (ICALP'99)*, volume 1644 of *Lecture Notes in Computer Science*, pages 422–432. Springer-Verlag, 1999.

[HR99b] Yoram Hirshfeld and Alexander Moshe Rabinovich. Quantitative temporal logic. In Jörg Flum and Mario Rodríguez-Artalejo, editors, *Proceedings of the 13th International Workshop on Computer Science Logic (CSL'99)*, volume 1683 of *Lecture Notes in Computer Science*, pages 172–187. Springer-Verlag, 1999.

[HR03] Yoram Hirshfeld and Alexander Moshe Rabinovich. Future temporal logic needs infinitely many modalities. *Information and Computation*, 187(2):196–208, 2003.

[HR04] Yoram Hirshfeld and Alexander Moshe Rabinovich. Logics for real time: Decidability and complexity. *Fundamenta Informaticae*, 62(1):1–28, 2004.

[HR06] Yoram Hirshfeld and Alexander Moshe Rabinovich. Expressiveness of metric modalities for continuous time. In Dima Grigoriev, John Harrison, and Edward A. Hirsch, editors, *Proceedings of the 1st International Computer Science Symposium in Russia: Computer Science – Theory and Applications (CSR'06)*, volume 3967 of *Lecture Notes in Computer Science*, pages 211–220. Springer-Verlag, 2006.

[HRS98] Thomas A. Henzinger, Jean-François Raskin, and Pierre-Yves Schobbens. The regular real-time languages. In Kim Guldstrand Larsen, Sven Skyum, and Glynn Winskel, editors, *Proceedings of the 25th International Colloquium on Automata, Languages and Programming (ICALP'98)*, volume 1443 of *Lecture Notes in Computer Science*, pages 580–591. Springer-Verlag, 1998.

[Kam68] Johan Anthony Willem Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, 1968.

[Koy90] Ron Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990.

[KP05] Yonit Kesten and Amir Pnueli. A compositional approach to CTL* verification. *Theoretical Computer Science*, 331(2–3):397–428, 2005.

[KS05] Antonín Kučera and Jan Strejček. The stuttering principle revisited. *Acta Informatica*, 41(7/8):415–434, 2005.

[Lam83] Leslie Lamport. What good is temporal logic? In R. E. A. Mason, editor, *Proceedings of the IFIP 9th World Computer Congress on Information Processing (IFIP'83)*, pages 657–668. North Holland/IFIP, 1983.

[LW05] Slawomir Lasota and Igor Walukiewicz. Alternating timed automata. In Vladimiro Sassone, editor, *Proceeding of the 8th International Conference on Foundations of Software Science and Computational Structures (FOSSACS'05)*, volume 3441 of *Lecture Notes in Computer Science*, pages 250–265. Springer-Verlag, 2005.

[MNP05] Oded Maler, Dejan Nickovic, and Amir Pnueli. Real time temporal logic: Past, present, future. In Paul Petterson and Wang Yi, editors, *Proceedings of the 3rd International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS'05)*, volume 3829 of *Lecture Notes in Computer Science*, pages 2–16. Springer-Verlag, 2005.

[MNP06] Oded Maler, Dejan Nickovic, and Amir Pnueli. From MITL to timed automata. In Eugene Asarin and Patricia Bouyer, editors, *Proceedings of the 4th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS'06)*, volume 4202 of *Lecture Notes in Computer Science*, pages 274–289. Springer-Verlag, 2006.

[MP72] Robert McNaughton and Seymour A. Papert. *Counter-Free Automata*, volume 65 of *MIT Research Monograph*. MIT Press, 1972.

[MP93] Zohar Manna and Amir Pnueli. Models for reactivity. *Acta Informatica*, 30(2):609–678, 1993.

[MP95] Oded Maler and Amir Pnueli. Timing analysis of asynchronous circuits using timed automata. In Paolo Camurati and Hans Eveking, editors, *Proceedings of the Advanced Research Working Conference on Correct Hardware Design and Verification Methods*, volume 987 of *Lecture Notes in Computer Science*, pages 189–205. Springer-Verlag, 1995.

[OW04] Joël Ouaknine and James Worrell. On the language inclusion problem for timed automata: Closing a decidability gap. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science (LICS'04)*, pages 54–63. IEEE Computer Society Press, 2004.

[OW05] Joël Ouaknine and James Worrell. On the decidability of metric temporal logic. In *Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science (LICS'05)*, pages 188–197. IEEE Computer Society Press, 2005.

[OW06] Joël Ouaknine and James Worrell. On metric temporal logic and faulty Turing machines. In Luca Aceto and Anna Ingólfsdóttir, editors, *Proceedings of the 9th International Conference Foundations of Software Science and Computation Structures (FoSSaCS'06)*, volume 3921 of *Lecture Notes in Computer Science*, pages 217–230. Springer-Verlag, 2006.

[PD06] Pavithra Prabhakar and Deepak D'Souza. On the expressiveness of MTL with past operators. In Eugene Asarin and Patricia Bouyer, editors, *Proceedings of the 4th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS'06)*, volume 4202 of *Lecture Notes in Computer Science*, pages 322–336. Springer-Verlag, 2006.

[PW97] Doron Peled and Thomas Wilke. Stutter-invariant temporal properties are expressible without the next-time operator. *Information Processing Letters*, 63(5):243–246, 1997.

[Rab03] Alexander Moshe Rabinovich. Automata over continuous time. *Theoretical Computer Science*, 300(1–3):331–363, 2003.

[RS97] Jean-François Raskin and Pierre-Yves Schobbens. State clock logic: A decidable real-time logic. In Oded Maler, editor, *Proceedings of the International Workshop on Hybrid and Real-Time Systems*, volume 1201 of *Lecture Notes in Computer Science*, pages 33–47. Springer-Verlag, 1997.

[RT97] Alexander Moshe Rabinovich and Boris A. Trakhtenbrot. From finite automata toward hybrid systems. In Ludwik Czaja Bogdan S. Chlebus, editor, *Proceedings of the 11th International Symposium on Fundamentals of Computation Theory (FCT'97)*, volume 1279 of *Lecture Notes in Computer Science*, pages 411–422. Springer-Verlag, 1997.

[Sch02] Ph. Schnoebelen. Verifying lossy channel systems has nonprimitive recursive complexity. *Information Processing Letters*, 83(5):251–261, 2002.

[Tra95] Boris A. Trakhtenbrot. Origins and metamorphoses of the trinity: Logics, nets, automata. In *Proceedings, 10th Annual IEEE Symposium on Logic in Computer Science*, pages 506–507. IEEE Computer Society Press, 1995.

[TW04] Denis Thérien and Thomas Wilke. Nesting until and since in linear temporal logic. *Theory of Computing Systems*, 37(1):111–131, 2004.

[Var97] Moshe Y. Vardi. Alternating automata: Unifying truth and validity checking for temporal logics. In William McCune, editor, *Proceedings of the 14th International Conference on Automated Deduction (CADE'97)*, volume 1249 of *Lecture Notes in Computer Science*, pages 191–206. Springer-Verlag, 1997.

[Wil94] Thomas Wilke. Specifying timed state sequences in powerful decidable logics and timed automata. In Hans Langmaack, Willem P. de Roever, and Jan Vytopil, editors, *Proceedings of the 3rd International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'94)*, volume 863 of *Lecture Notes in Computer Science*, pages 694–715. Springer-Verlag, 1994.

[Wil99] Thomas Wilke. Classifying discrete temporal properties. In Christoph Meinel and Sophie Tison, editors, *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS'99)*, volume 1563 of *Lecture Notes in Computer Science*, pages 32–46. Springer-Verlag, 1999.

[Wol83] Pierre Wolper. Temporal logic can be more expressive. *Information and Control*, 56(1):72–99, 1983.

[WVS83] Pierre Wolper, Moshe Y. Vardi, and A. Prasad Sistla. Reasoning about infinite computation paths. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science (FOCS'83)*, pages 185–194. IEEE Computer Society Press, 1983.