Equivalence of the Pointwise and Continuous Semantics of First Order Logic with Linear Constraints

Deepak D'Souza
Indian Institute of Science
Bangalore 560012, India.
deepakd@csa.iisc.ernet.in

Raveendra Holla
Indian Institute of Science
Bangalore 560012, India.
raveendra@csa.iisc.ernet.in

Raj Mohan Matteplackel
Indian Institute of Science
Bangalore 560012, India.
raj@csa.iisc.ernet.in

Abstract

We consider a first-order logic with linear constraints which can be interpreted naturally in both a pointwise or continuous way over timed words. We show that the two interpretations of this logic coincide in terms of expressiveness. As a consequence it follows that the pointwise and continuous semantics of the logic TPTL *with the since operator [2] also coincide. We also exhibit a normal form for formulae in these logics.*

1 Introduction

Several real-time logics proposed in the literature have been interpreted over timed behaviours in two natural ways which have come to be called the "pointwise" and "continuous" interpretations. These interpretations can be carried out over the popular model of a timed word which is a sequence of actions along with their associated real-valued time stamps. In the pointwise semantics formulae may be asserted only at the discrete set of points where actions occur (the so called "action points"), while in the continuous semantics formulae may be asserted at arbitrary time points. To illustrate these semantics, consider the popular timed temporal logic Metric Temporal Logic (MTL) [5, 1, 6], which extends the U operator of classical LTL with an interval index to allow formulae of the form $\theta U_I \eta$, which says that θ is satisfied until a future time point where η is satisfied, and which lies at a distance that falls in the interval I. Consider a timed word σ shown in Fig. 1 in which the first action is an a at time 2, followed subsequently by only b's. The MTL formula $\Diamond(\Diamond_{[1,1]} a)$ (where $\Diamond_I \theta$ stands for $\top U_I \theta$) asserts that there is a time point t from which an action a occurs in the future at a distance of 1. The formula is satisfied in the timed word σ in the continuous semantics, but *not* in the pointwise semantics (essentially since there is no action at time point 1).

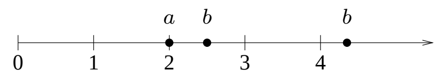


Figure 1. Timed word σ

The Timed Temporal Logic (TPTL) of Alur and Henzinger [2] is a well-known timed temporal logic for specifying real-time behaviours. The logic is interpreted over timed words and extends classical LTL with the "freeze" quantifier $x.\theta$ which binds x to the value of the current time point, along with the ability to constrain these time points using linear constraints of the form $x \sim y + c$. For example the formula $x.(\Diamond y.(a \wedge y = x + 2))$ says that with respect to the current time point, an a-event occurs exactly two time units later. Once again, the $\Diamond$ operator can be interpreted in a pointwise or continuous manner.

It is not difficult to see that for a typical timed logic the continuous semantics is at least as expressive as the pointwise one, since one can ask for a time point to be an action point by asserting $\bigvee_{a \in \Sigma} a$ (where Σ is the set of possible actions) at each quantified time point. For the logic MTL and its variants there have been several results in the literature which show that the continuous semantics is in fact strictly more expressive than the pointwise one [3, 4, 7]. Thus the logics MTL, MTL_S (MTL with the "since" operator S), MTL_{S_I} (MTL with the S_I operator), and MITL (MTL restricted to non-singular intervals), are all strictly more expressive in the continuous semantics than their pointwise counterparts.

In this paper we consider the expressiveness of the pointwise and continuous interpretations of a natural first-order logic with linear constraints which is interpreted over timed words and is similar in flavour to TPTL. Thus the logic allows atomic predicates of the form $a(x)$ which says that an a-event occurs at time point x, and constraints of the form $x \sim y + c$. The interpretation of the quantifier $\exists x$ depends on the continuous or pointwise semantics: in the continuous semantics it is interpreted as "there exists a time point x," while in the pointwise semantics it is interpreted as "there exists an *action point* x." As an example, the $\mathrm{FO}(<,+)$ formula $\exists x(\exists y(a(y) \wedge y = x + 1))$ is satisfied in the example timed word σ above in the continuous interpretation while it is not satisfied in the pointwise semantics.

Our main result in this paper is that the expressiveness of the logic in these two semantics coincide. The logic $\mathrm{FO}(<,+)$ can be seen to be expressively the same as TPTL with the "since" modality, which we denote TPTL_S. Thus our expressive equivalence result carries over to TPTL_S as well. To our knowledge these are the first instances of real-time logics for which the pointwise and continuous semantics are known to be equally expressive.

The main proof idea is to first go over to a normal form for $\mathrm{FO}(<,+)$ sentences in which all quantified subformulae are essentially of the form $\exists x(a(x) \wedge \pi \wedge \nu)$, where a is an action in Σ, π is a linear constraint involving x, and ν is a conjunction of formulae satisfying a similar restriction on quantified subformulae. This is similar in spirit to a normal form proposed for a sublogic of TPTL in [3]. We then show how to transform an arbitrary sentence in normal form in $\mathrm{FO}(<,+)$ to a sentence in $\mathrm{FO}(<,+)$ which uses only "active" quantifiers. We say that the formula $\exists x\varphi$ is actively quantified if φ is of the form $a(x) \wedge \psi$ for some action a and formula ψ; and say it is "passively" quantified otherwise. A sentence in which all quantifiers are active, is clearly equivalent to a pointwise formula. Thus, we show how to eliminate passively quantified variables using only actively quantified ones. As an example, the formula $\exists x(1 \leq x \wedge \exists y(b(y) \wedge x + 1 \leq y))$ is passively quantified. We can eliminate x from it to get the equivalent actively quantified formula $\exists y(b(y) \wedge 2 \leq y)$.

2 Preliminaries

Let $\mathbb{N}$ denote the set of non-negative integers, and $\mathbb{R}$ and $\mathbb{R}_{\geq 0}$ the set of reals and non-negative reals respectively. By an interval we will mean a convex subset of $\mathbb{R}$, and denote these using standard notation: for example $[1, 3)$ will denote the set $\{t \in \mathbb{R} \mid 1 \leq t < 3\}$.

For an alphabet A we denote by A^ω the set of infinite words over A. Let Σ be a finite alphabet of actions, which we fix for the rest of this paper. An (infinite) *timed word* σ over Σ is an element of $(\Sigma \times \mathbb{R}_{\geq 0})^\omega$ of the form $(a_0, t_0)(a_1, t_1)\cdots$, satisfying the conditions that: for each $i \in \mathbb{N}$, $t_i < t_{i+1}$ (*strict monotonicity*), and for each $t \in \mathbb{R}_{\geq 0}$ there exists an $i \in \mathbb{N}$ such that $t < t_i$ (*progressiveness*). For convenience, we will also assume in this paper that $t_0 = 0$. We will sometimes represent the timed word σ above as a pair (α, τ), where $\alpha = a_0a_1\cdots$ and $\tau = t_0t_1\cdots$. Thus $\alpha(i)$ and $\tau(i)$ denote the action and the time stamp respectively, in σ at position i. We write $T\Sigma^\omega$ to denote the set of all timed words over Σ.

We now introduce the linear constraints we use in this paper, and some notation for manipulating them. We assume a supply of variables $Var = \{x, y, \ldots\}$ which we will use in constraints as well as later in our logics. We use restricted linear constraints of the form $x \sim y + c$ or $x \sim c$, where x and y are variables in Var, $\sim$ is one of the relations in $\{<, \leq, =, \geq, >\}$, and $c \in \mathbb{R} \cup \{\infty\}$. We call these constraints *simple constraints*. In general, we will allow constraints to be boolean combinations of simple constraints given by the syntax $\delta ::= g \mid \neg\delta \mid \delta\wedge\delta \mid \delta\vee\delta$, where g is a simple constraint.

An *interpretation* for variables is a map $\mathbb{I} : Var \to \mathbb{R}$. For $t \in \mathbb{R}$ and $x \in Var$ we will use $\mathbb{I}[t/x]$ to represent the interpretation which sends x to t, and agrees with $\mathbb{I}$ on all other variables. When we are interested in a finite set of variables $\{x_1, \ldots, x_n\}$ we will write $[t_1/x_1, \ldots, t_n/x_n]$ to represent an interpretation that maps each x_i to t_i. For an interpretation $\mathbb{I}$ and a constraint δ, we write $\mathbb{I} \models \delta$ to mean that the constraint δ is satisfied in the interpretation $\mathbb{I}$, and define it in the expected way.

We will use the notion of an "interval constraint" for a variable x, which is a constraint whose solution set for x given any interpretation for variables other than x, forms an interval. More precisely, an *interval constraint* for x is a constraint of the form $\pi_l \wedge \pi_r$, where π_l is a positive boolean combination of simple constraints of the form $e \prec x$ and π_r is a positive boolean combination of simple constraints of the form $x \prec e$, where $\prec$ stands for one of the relations $<$ or $\leq$. We call π_l the "left boundary" of π, and denote it by $l(\pi)$, and π_r the "right boundary" of π and denote it by $r(\pi)$. It will be convenient to view the left boundary of an interval constraint as a disjunction of conjunctions of simple constraints, or equivalently as the "minimum of maximums" of each conjunct. Note that if π_1 and π_2 are two interval constraints, then $\pi_1 \wedge \pi_2$ is also an interval constraint, as it can be written as $(l(\pi_1) \wedge l(\pi_2)) \wedge (r(\pi_1) \wedge r(\pi_2))$.

Let π_1 and π_2 be two interval constraints for x. Then the constraint $\pi_3 = \neg r(\pi_1) \wedge \neg l(\pi_2)$ is again an interval which spans from the right boundary of π_1 to the left boundary of π_2. Note here that negation of right boundary of π_1 becomes a left boundary of π_3 which begins from where interval π_1 ends. If for some interpretation the intervals π_1 and π_2 are overlapping or the interval π_2 precedes π_1, then this interval will be empty. This is useful in forcing some ordering over intervals. For example consider a formula (from a generic first-order logic interpreted over the reals) of the form

$$\exists y(\varphi \wedge \exists z\psi)$$

Let $\pi_1 = y-1 \leq x \leq y+1$ and $\pi_2 = z-2 \leq x \leq z+2$ be two interval constraints for x. Each value for y and z gives us an interval of values for x satisfying π_1 (and similarly for π_2). If we want to restrict ourselves to values of y and

z such that the induced interval for x in π_1 comes strictly "before" that of π_2, we can modify the formula as follows:

$$\begin{array}{ll} & \exists y(\varphi \wedge \exists z(\psi \wedge \exists x(\neg r(\pi_1) \wedge \neg l(\pi_2)))) \\ = & \exists y(\varphi \wedge \exists z(\psi \wedge \exists x(y+1 < x \wedge x < z-2))) \end{array}$$

In a similar manner, the interval constraint $l(\pi_1) \wedge \neg l(\pi_2)$ will be nonempty (i.e. have a nonempty solution set) iff the interval induced by π_1 starts strictly before the interval induced by π_2, and the interval constraint $\neg r(\pi_1) \wedge r(\pi_2)$ will be nonempty iff the interval π_1 ends strictly before the interval π_2.

As a final piece of notation, we recall the well-known Fourier-Motzkin method (see [8]) for eliminating variables from constraints. Consider a conjunction of simple constraints π. We assume for now that π has no strict constraints. The constraints in π can then be written as:

$$\begin{array}{lll} e_i & \leq & x \quad (i \in \{1,\ldots,m_1\}) \\ x & \leq & e_j \quad (j \in \{m_1+1,\ldots,m_2\}) \\ e_k & \leq & f_k \quad (k \in \{m_2+1,\ldots,m_3\}), \end{array}$$

where $0 \leq m_1 \leq m_2 \leq m_3$, and each e_i and f_i are expressions not containing x. The variable x can be eliminated by taking the conjunction of the simple constraints below which do not contain x:

$$e_i \leq e_j \quad (i \in \{1,\ldots,m_1\}, j \in \{m_1+1,\ldots,m_2\}) \qquad (1)$$
$$e_k \leq f_k \quad (k \in \{m_2+1,\ldots,m_3\}). \qquad (2)$$

We denote this conjunction by $fm_x(\pi)$. The constraint $fm_x(\pi)$ preserves the solution set of π in that it is the projection of the solution set of π to the variables in π other than x. More precisely, if $x, y_1, \ldots, y_n$ were the variables in π, then $[t_1/y_1, \ldots, t_n/y_n]$ is a solution for $fm_x(\pi)$ iff there exists $t \in \mathbb{R}$ such that $[t/x, t_1/y_1, \ldots, t_n/y_n]$ is a solution to π.

The method can be extended to the case when some of the constraints are strict, by writing $e_i < e_j$ in (1) above whenever at least one of the i-th or j-th constraints is strict, and $e_i \leq e_j$ otherwise. It can also be extended to eliminate x from an interval constraint π for x. For this we first rewrite $l(\pi)$ and $r(\pi)$ as a disjunction of conjuncts δ_i and ρ_i respectively, and then taking the disjunction of each $fm_x(\delta_i \wedge \rho_j)$.

3 The logic $\mathrm{FO}(<,+)$

We now introduce the first-order logic of linear constraints we will be concerned with in this paper. The logic is denoted $\mathrm{FO}(<,+)$ and its formulae are given by the following syntax:

$$\varphi ::= a(x) \mid g \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid \exists x \varphi,$$

where $a \in \Sigma$, $x \in \mathit{Var}$, and g is a simple constraint.

The logic is interpreted over timed words over Σ. We first define the *continuous* semantics for $\mathrm{FO}(<,+)$. Let φ be a formula in $\mathrm{FO}(<,+)$. Let $\sigma = (\alpha, \tau)$ be a timed word over Σ, and let $\mathbb{I}$ be an interpretation for variables. Then the satisfaction relation $\sigma, \mathbb{I} \models_c \varphi$ (read "σ satisfies φ in the interpretation $\mathbb{I}$ in the continuous semantics") is inductively defined as:

$$\begin{array}{lll} \sigma, \mathbb{I} \models_c a(x) & \text{iff} & \exists i \in \mathbb{N}: \ \alpha(i) = a \text{ and } \tau(i) = \mathbb{I}(x) \\ \sigma, \mathbb{I} \models_c g & \text{iff} & \mathbb{I} \models g \\ \sigma, \mathbb{I} \models_c \neg\psi & \text{iff} & \sigma, \mathbb{I} \not\models_c \psi \\ \sigma, \mathbb{I} \models_c \psi \wedge \mu & \text{iff} & \sigma, \mathbb{I} \models_c \psi \text{ and } \sigma, \mathbb{I} \models_c \mu \\ \sigma, \mathbb{I} \models_c \psi \vee \mu & \text{iff} & \sigma, \mathbb{I} \models_c \psi \text{ or } \sigma, \mathbb{I} \models_c \mu \\ \sigma, \mathbb{I} \models_c \exists x \psi & \text{iff} & \exists t \in \mathbb{R}_{\geq 0}: \ \sigma, \mathbb{I}[t/x] \models_c \psi. \end{array}$$

We say that formulae φ and ψ are (logically) *equivalent*, denoted $\varphi \equiv \psi$, if for every timed word σ and for every interpretation $\mathbb{I}$, we have $\sigma, \mathbb{I} \models_c \varphi$ iff $\sigma, \mathbb{I} \models_c \psi$.

A variable x is said to occur *free* in a formula φ if it has an occurrence outside the scope of a quantifier $\exists x$. A *sentence* is a formula in which there are no free occurrences of variables. Thus for a sentence φ, the interpretation plays no role, and we write the satisfaction relation as simply $\sigma \models \varphi$. The timed language defined by an $\mathrm{FO}(<,+)$ sentence φ in the continuous semantics is denoted $L^c(\varphi)$ and defined to be $\{\sigma \in T\Sigma^\omega \mid \sigma \models_c \varphi\}$. We denote this continuous version of the logic by $\mathrm{FO}^c(<,+)$.

In the *pointwise* version of the logic the quantification is over action points in the timed word. The satisfaction relation "$\models_{pw}$" for the pointwise semantics is defined as in the continuous case, except for the $\exists x$ clause which is defined as:

$$\sigma, \mathbb{I} \models_{pw} \exists x \psi \quad \text{iff} \quad \exists i \in \mathbb{N}: \ \sigma, \mathbb{I}[\tau(i)/x] \models_{pw} \psi.$$

The timed language defined by a sentence φ in the pointwise semantics is defined to be $L^{pw}(\varphi) = \{\sigma \in T\Sigma^\omega \mid \sigma \models_{pw} \varphi\}$. We denote the pointwise version of the logic by $\mathrm{FO}^{pw}(<,+)$.

Our aim in this paper is to show that the logics $\mathrm{FO}^c(<,+)$ and $\mathrm{FO}^{pw}(<,+)$ are expressively equivalent, in the sense that the class of timed languages definable by sentences in the two logics coincide.

4 A normal form for $\mathrm{FO}(<,+)$ sentences

In this section we exhibit a normal form for $\mathrm{FO}(<,+)$ sentences which will be useful in our proofs. An $\mathrm{FO}(<,+)$ formula is said to be in $\exists$-*normal form* if it is of the form

$$\exists x(a(x) \wedge \pi(x) \wedge \alpha),$$

where $a \in \Sigma$, $\pi(x)$ is an interval constraint for x, and α is a conjunction of formulae of the form ψ or $\neg\psi$, where each

ψ is again in $\exists$-normal form. In addition, we allow any of the components $a(x)$ or α to be absent. We assume that all the variables are distinct.

The figure below shows an the formula tree of an example formula in $\exists$-normal form, and the structure of a formula tree in normal form.

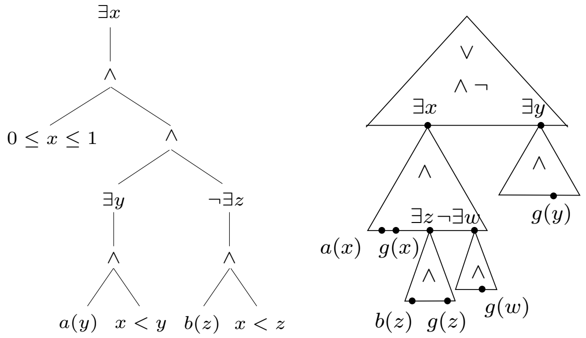


Figure 2. Formula tree of a formula in $\exists$-normal form, and (right) a depiction of a formula tree in normal form.

Theorem 1 *Every* $\mathrm{FO}(<,+)$ *sentence can be transformed to an equivalent sentence in* $\exists$*-normal form.*

Proof Let φ be an $\mathrm{FO}(<,+)$ sentence. Since φ is a sentence it must be a boolean combination of sentences of the form $\exists x \varphi'$. We transform φ into an equivalent sentence in $\exists$-normal form by carry out the following steps on its formula tree:

1. In every subtree rooted at a $\exists$ node, push every $\neg$ operator downwards over $\vee$ and $\wedge$ nodes. We stop pushing when we reach a $\exists$ node or an "action" node of the form $a(x)$. However we push negations all the way through the "g" nodes. After this step, the subtree below every $\exists$ node contains only conjunctions and disjunctions of a, $\neg a$, $\exists$, $\neg\exists$, and g nodes.

2. In this step and the next we view $\neg\exists$ as a single composite node in the formula tree. Pull all the $\vee$'s upwards in the formula tree using the following identities: $\alpha_1 \wedge (\alpha_2 \vee \alpha_3) \equiv (\alpha_1 \wedge \alpha_2) \vee (\alpha_1 \wedge \alpha_3)$, $\exists x(\alpha_1 \vee \alpha_2) \equiv (\exists x \alpha_1) \vee (\exists x \alpha_2)$, and $\neg\exists x(\alpha_1 \vee \alpha_2) \equiv (\neg\exists x(\alpha_1)) \wedge (\neg\exists x(\alpha_2))$. Using these identities we can propagate all the $\vee$ nodes upwards so that each subtree rooted at a $\exists$ node or $\neg\exists$ node contains only conjunctions of a, $\neg a$, $\exists$, $\neg\exists$, and g nodes. It is not difficult to see that this step terminates (see the Appendix for an argument).

3. In this step we pull up from a subtree rooted at a $\exists x$ node, all nodes which are independent of x, namely nodes of the form $b(y)$, $\neg b(y)$ (with $y \neq x$), and g where g does not contain x. This is done by repeatedly applying the following equivalences starting from the lower most $\exists x$ or $\neg\exists x$ nodes: $\exists x(b(y) \wedge \alpha) \equiv b(y) \wedge \exists x(\alpha)$ and $\neg\exists x(b(y) \wedge \alpha) \equiv \neg b(y) \vee \neg\exists x(\alpha)$ (when $x \neq y$). We can use similar equivalences for $\neg b(y)$ and g to pull them up the tree. Finally, we move all the newly generated $\vee$'s up the tree using Step 2.

   After this step, the subtrees rooted at each $\exists x$ node is a conjunction of $a(x)$, $\neg a(x)$, $\exists$, $\neg\exists$ and $g(x)$ nodes.

4. We now update the formula tree with the following equivalences: $a(x) \wedge b(x) \equiv \perp$, where $\perp$ is the formula $0 < 0$ denoting "false," and $a(x) \wedge \neg b(x) \equiv a(x)$, whenever $a, b \in \Sigma$ with $a \neq b$. After this step the only action-related nodes in a subtree rooted at a $\exists x$ node are a single action node $a(x)$ or a conjunction of negation of actions of the form $\bigwedge_{a \in A} \neg a(x)$ for some $A \subseteq \Sigma$.

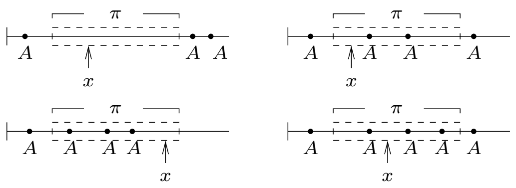
5. We now replace subtrees rooted at $\exists x$ nodes which contain formulae of the form $\exists x(\bigwedge_{a \in A} \neg a(x) \wedge \pi \wedge \alpha)$, by a disjunction of $\exists$ and $\neg\exists$ subformulae which contain at most one action in their "immediate" subtree. This is described below. We can now pull the newly generated $\vee$ nodes up the formula tree using Step (2). After this step the subtree rooted at every $\exists x$ node contains only conjunctions of $a(x)$, $\exists$, $\neg\exists$, and $g(x)$ nodes. We can then collect the $g(x)$ nodes together to get a single conjunction of constraints $\pi(x)$. Thus finally each subtree rooted at a $\exists$ node is in $\exists$-normal form, and the resulting formula is in $\exists$-normal form.

We now show how we can replace a formula φ of the form $\exists x(\bigwedge_{a \in A} \neg a(x) \wedge \pi(x) \wedge \alpha)$ by a disjunction of formulae in $\exists$-normal form. Let $A(x)$ be shorthand for the formula $\bigvee_{a \in A} a(x)$. Then $\varphi = \exists x(\neg A(x) \wedge \pi(x) \wedge \alpha)$.

We claim that $\varphi \equiv \psi_1 \vee \psi_2 \vee \psi_3 \vee \psi_4$ where:

$$\begin{aligned}
\psi_1 = \quad & \neg\exists x(A(x) \wedge \pi(x)) \wedge \exists x(\pi(x) \ \wedge \ \alpha) \\
\psi_2 = \quad & \exists x_l(A(x_l) \wedge \pi[x_l/x] \wedge \neg\exists x'(A(x') \wedge \pi[x'/x] \\
& \wedge x' < x_l) \wedge \exists x(\pi(x) \wedge x < x_l \ \wedge \ \alpha)) \\
\psi_3 = \quad & \exists x_l(A(x_l) \wedge \pi[x_l/x] \wedge \exists x_r(A(x_r) \wedge \pi[x_r/x] \\
& \wedge \neg\exists x'(A(x') \wedge \pi[x'/x] \wedge x_l < x' < x_r) \\
& \wedge \exists x''(\pi[x''/x] \wedge x_l < x'' < x_r \ \wedge \ \alpha))) \\
\psi_4 = \quad & \exists x_r(A(x_r) \wedge \pi[x_r/x] \wedge \neg\exists x'(A(x') \wedge \pi[x'/x] \\
& \wedge x_r < x') \wedge \exists x''(\pi[x''/x] \wedge x_r < x'' \ \wedge \ \alpha))
\end{aligned}$$

This translation can be justified using the figure below which shows the four cases corresponding to each $\psi_1, \psi_2, \psi_3, \psi_4$ in clockwise order starting from the top left.



The formula φ is satisfied in a given timed word and interpretation iff either there is no A-point in π and there is a point in π satisfying α (ψ_1 is satisfied), or there is a point in π satisfying α that occurs before the first A-point in π (ψ_2 is satisfied), or between two consecutive A-points in π (ψ_3 is satisfied), or after the last A-point in π (ψ_4 is satisfied). We note that we make use of the progressiveness assumption on timed words here: since the action timepoints are progressive, the set of A-points cannot be dense in π. □

5 Expressive equivalence of FO^c and FO^{pw}

In this section our aim is to show that the logics $\mathrm{FO}^{pw}(<,+)$ and $\mathrm{FO}^c(<,+)$ are expressively equivalent. It is easy to translate an $\mathrm{FO}^{pw}(<,+)$ sentence φ to an equivalent $\mathrm{FO}^c(<,+)$ sentence by simply replacing every $\exists x\psi$ subformula, by $\exists x(\bigvee_{a\in\Sigma} a(x) \wedge \psi')$, where ψ' is obtained by similarly replacing $\exists$ subformulae in ψ.

In the converse direction, let us call an $\mathrm{FO}^c(<,+)$ formula φ *actively quantified* (or simply *active*) if every $\exists$ subformula is of the form $\exists x(a(x)\wedge\varphi')$, for some action $a \in \Sigma$ and formula φ'. An active $\mathrm{FO}^c(<,+)$ sentence clearly defines the same language of timed words, regardless of the semantics being pointwise or continuous. Hence, our aim in the rest of this section is to show how we can go from an arbitrary formula in $\mathrm{FO}^c(<,+)$ to an equivalent active formula.

An arbitrary formula in the continuous semantics has the advantage of being able to associate any value in $\mathbb{R}_{\geq 0}$ to its variables, whereas an actively quantified variable can be asserted only at the action points in a timed word. As an example, consider the language of all timed words over $\{a, b\}$, where there is a b in the interval $[1, 2]$ for which there is no a in the interval $[0, 1]$ which is exactly a distance of one time unit from it. This language can be expressed in FO^c as:

$$\exists x(\neg a(x) \wedge 0 \leq x \wedge x \leq 1 \wedge \exists y(b(y) \wedge y = x + 1)).$$

The same formula in the pointwise semantics however defines only a strict subset of the above language. One can nonetheless give an equivalent formula in the pointwise semantics, namely

$$\exists y(b(y)\wedge 1 \leq y \leq 2\wedge\neg\exists x(a(x)\wedge 0 \leq x \leq 1\wedge y = x+1)).$$

However we need a systematic way of doing this. The method we propose is to first convert the given formula to normal form, and then essentially replace passively quantified subformulae by equivalent actively quantified ones.

We will formally define our translation in Lemma 1. We first illustrate the main ideas behind the translation using a sequence of examples of increasing complexity.

In the first example x is passively quantified:

$$\exists x\big(0 \leq x \leq 1 \wedge \exists y(a(y) \wedge x + 1 \leq y \leq x + 1.2)\big). \quad (3)$$

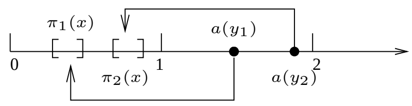


Figure 3. Intervals of x where formula (3) is satisfied.

The above formula is true iff there is a point in $[0, 1]$, from which there is an action a at a distance which lies in $[1, 1.2]$. We can rewrite the formula as follows:

$$\exists y(a(y) \wedge \exists x(0 \leq x \wedge y - 1.2 \leq x \leq 1 \wedge x \leq y - 1)). \quad (4)$$

Let $\pi(x)$ be the constraint $0 \leq x \wedge y - 1.2 \leq x \leq 1 \wedge x \leq y - 1$. We can obtain an active formula equivalent to the formula (4) above:

$$\exists y(a(y) \wedge 1 \leq y \leq 2.2), \quad (5)$$

by using the FM-elimination method to eliminate x from the interval constraint $\pi(x)$. Observe that if (5) is satisfied at some $y = t$, then every point in the interval $\pi[t/y]$ will satisfy (3). If $y = 1.5$ then the interval of x will be $[0.3, 0.5]$ and if $y = 2.1$ then interval of x will be $[0.9, 1]$.

As the second example, consider the following modified version of (3):

$$\exists x\big(0 \leq x \leq 1 \wedge \neg\exists y(a(y) \wedge x + 1 \leq y \leq x + 1.2)\big) \quad (6)$$

We can make use of our elimination technique for formula (3) to eliminate the passively quantified variable x from this formula. We first consider all "intervals" of x in [0,1] which satisfy (3) in the given model (the bracketed region in the figure). The formula (6) is true in that model iff there are some "gaps" in $[0, 1]$ where formula (3) is not satisfied. There can be the following four types of gaps:

1. No x satisfies (3), so the whole of $[0, 1]$ is a gap.
2. A gap from 0 to the beginning of the "first" interval of x satisfying (3).
3. A gap between two "consecutive" intervals of x satisfying formula (3).
4. A gap from the end of the "last" interval of x satisfying (3) to 1.

The formula (6) can be satisfied iff any of the above gaps exist and these gaps define the intervals of x that satisfy (6). We will now give four formulae ψ_1, ψ_2, ψ_3 and ψ_4 which characterise the four cases above.

For the first case, there must be no a's in the interval from $[1, 2.2]$. The formula ψ_1 below says this:

$$\neg\exists y(a(y) \wedge 1 \leq y \leq 2.2). \tag{7}$$

The corresponding interval of values of x which satisfy (6) is $0 \leq x \leq 1$.

For case 2 we need to characterise the first interval for x where formula (3) is true. Let y_1 be a value for y which satisfies formula (5) in a given model. Let us denote the corresponding interval of x generated by y_1 by $\pi_1(x) = \pi[y_1/y]$. If $\pi_1(x)$ is the first interval of x satisfying (3), then there must not exist any value y_2 for y such that it satisfies formula (5) and the left boundary of its interval $\pi_2(x) = \pi[y_2/y]$ starts before π_1. This can be stated as follows:

$$\exists y_1\Big(a(y_1) \wedge 1 \leq y_1 \leq 2.2$$
$$\wedge\neg\exists y_2\big(a(y_2) \wedge 1 \leq y_2 \leq 2.2 \wedge \exists x(l(\pi_2) \wedge \neg l(\pi_1))\big)\Big).$$

The formula $\exists x(l(\pi_2)\wedge\neg l(\pi_1))$ evaluates to true iff the left boundary of π_2 starts before π_1. The variable x present here can be eliminated using FM-elimination to get an active formula. If the above formula is satisfied, then every point in the interval of $0 \leq x \wedge \neg l(\pi_1)$ will satisfy (6) as in this interval the formula (3) is not satisfied (because π_1 is the first interval satisfying the formula (3)). One issue still remains: if π_1 starts from 0 itself, then even if the above formula is satisfied, the interval of points satisfying (6) may be empty. Incorporating this condition we get following formula:

$$\exists y_1\Big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge fm_x(0 \leq x \wedge \neg l(\pi_1))$$
$$\wedge\neg\exists y_2\big(a(y_2) \wedge 1 \leq y_2 \leq 2.2 \wedge fm_x(l(\pi_2) \wedge \neg l(\pi_1))\big)\Big)$$

Here condition $fm_x(0 \leq x\wedge\neg l(\pi_1))$ ensures that, the interval from 0 to the left boundary of π_1 is not empty. Finally eliminating x using FM-elimination, we get the required formula ψ_2:

$$\begin{aligned}&\exists y_1\big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge 1.2 < y_1\\ &\wedge\neg\exists y_2(a(y_2) \wedge 1 \leq y_2 \leq 2.2 \wedge y_2 < y_1)\big)\end{aligned} \tag{8}$$

If any timed word satisfies the formula (8), then it satisfies the formula (6) for all points in the interval $0 \leq x < y_1 - 1.2$.

Let $y = y_1$ and $y = y_2$ are satisfying (5) in a given model and the interval of x generated by y_1 and y_2 be $\pi_1(x)$ and $\pi_2(x)$. If $\pi_1(x)$ and $\pi_2(x)$ are disjoint and consecutive intervals then the interval $\neg r(\pi_1)\wedge\neg l(\pi_2)$ must be nonempty and there must not exist any instance of $y = y_3$ such that it satisfy (5) and the interval π_3 has any overlapping with the interval $\neg r(\pi_1) \wedge \neg l(\pi_2)$ This can be stated in the formula as follows:

$$\exists y_1\Big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge \exists y_2\Big(a(y_2) \wedge 1 \leq y_2 \leq 2.2$$
$$\wedge\exists x(\neg r(\pi_1) \wedge \neg l(\pi_2))\wedge$$
$$\neg\exists y_3\big(a(y_3) \wedge 1 \leq y_3 \leq 2.2 \wedge \exists x(\pi_3 \wedge \neg r(\pi_1) \wedge \neg l(\pi_2))\big)\Big)\Big)$$

The x present here can be eliminated using FM-elimination technique to get an active formula. If the above formula is satisfied, then every point in the interval of $\neg r(\pi_1)\wedge\neg l(\pi_2)$ satisfies (6) as in this interval (3) is not satisfied (because π_1 and π_2 are two consecutive intervals satisfying (3) and the gap between them can not satisfy (3)). Finally, eliminating x using FM-elimination, we get ψ_3

$$\begin{aligned}&\exists y_1\Big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge \exists y_2\big(a(y_2) \wedge 1 \leq y_2 \leq 2.2\\ &\wedge(y_1 + 0.2 < y_2)\\ &\wedge\neg\exists y_3(a(y_3) \wedge 1 \leq y_3 \leq 2.2 \wedge y_1 < y_3 < y_2)\big)\Big)\end{aligned} \tag{9}$$

If any timed word satisfies (9), then it satisfies (6) for all points in the interval $y_1 - 1 < x < y_2 - 1.2$.

Let $y = y_1$ is satisfying (5) in a given model and the interval of x generated by y_1 be $\pi_1(x)$. If $\pi_1(x)$ is the last interval of x satisfying (3), then there must not exist any instance of $y = y_2$ such that it satisfy (5) and the right boundary of its interval ends after π_1. Also we make sure that, the interval from the right boundary of π_1 to the right boundary of interval $[0, 1]$ is nonempty. This can be stated in the formula as follows:

$$\exists y_1\Big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge fm_x(\neg r(\pi_1) \wedge x \leq 1)$$
$$\wedge\neg\exists y_2\big(a(y_2) \wedge 1 \leq y_2 \leq 2.2 \wedge fm_x(\neg r(\pi_1) \wedge r(\pi_2))\big)\Big)$$

If the above formula is satisfied, then every point in the interval of $\neg r(\pi_1) \wedge x \leq 1$ will satisfy (6) as in this interval (3) is not satisfied (because π_1 is the last interval satisfying (3)). Finally eliminating x using FM-elimination, we get ψ_4

$$\begin{aligned}&\exists y_1\big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge y_1 < 2\\ &\wedge\neg\exists y_2(a(y_2) \wedge 1 \leq y_2 \leq 2.2 \wedge y_1 < y_2)\big)\end{aligned} \tag{10}$$

If any timed word satisfies (10), then it satisfies (6) for all points in the interval $y_1 - 1 < x \leq 1$.

Thus we can express (6) as the disjunction of active formulae (7), (8), (9) and (10). The disjuncts above do not contain the variable x. Nevertheless, for each ψ_i we have an interval constraint for x which gives us the values of x satisfying formula (6).

As the final example, consider the following modified version of (6):

$$\begin{aligned}&\exists x\big(0 \leq x \leq 1\wedge\\ &\neg\exists y(a(y) \wedge x + 1 \leq y \leq x + 1.2) \wedge \exists z(b(z) \wedge x + 1 \leq z)\big)\end{aligned} \tag{11}$$

Here x is again a passive variable, but there are multiple conjuncts under the scope of x. Let $\varphi_1 = \neg\exists y(a(y) \wedge x + 1 \leq y \leq x + 1.2)$ and $\varphi_2 = \exists z(b(z) \wedge z \geq x + 1)$. The above formula (11) is true iff there exists a point x from $[0, 1]$ such that both φ_1 and φ_2 are satisfied. But from the second example, we know exact intervals in $[0, 1]$ which satisfy φ_1. We further ask, whether φ_2 can be satisfied in the intervals where φ_1 is already satisfied. We know that when φ_1 is satisfied for $0 \leq x \leq 1$, one of the ψ_1, ψ_2, ψ_3 or ψ_4 is satisfied. If $\psi_1$is satisfied, then we check whether φ_2 can be satisfied in the interval $0 \leq x \leq 1$. That is:

$$\neg\exists y(a(y) \wedge 1 \leq y \leq 2.2) \wedge \exists x(0 \leq x \leq 1 \wedge \varphi_2))$$

Similarly, if ψ_2 is satisfied, then we check whether φ_2 can be satisfied in the interval $0 \leq x < y_1 - 1.2$. That is:

$$\begin{aligned}&\exists y_1\big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge 1.2 < y_1\\&\wedge\neg\exists y_2(a(y_2) \wedge 1 \leq y_2 \leq 2.2 \wedge y_2 < y_1)\\&\wedge\exists x(0 \leq x < y_1 - 1.2 \wedge \varphi_2)\big)\end{aligned}$$

and so on. Now the problem became much simpler as we need to eliminate x from formulae of the form $\exists x(\pi(x) \wedge \varphi_2)$, which is similar to the method we adopted for the first example. Eliminating x from all the four cases using FM elimination we get following formulae and their corresponding intervals:

1. $$\neg\exists y(a(y) \wedge 1 \leq y \leq 2.2) \wedge \exists z(b(z) \wedge 1 \leq z)$$

   The interval constraint for this is: $0 \leq x \wedge x \leq 1 \wedge x \leq z - 1$.

2. $$\begin{aligned}&\exists y_1\big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge 1.2 < y_1\\&\wedge\neg\exists y_2(a(y_2) \wedge 1 \leq y_2 \leq 2.2 \wedge y_2 < y_1)\\&\wedge\exists z(b(z) \wedge 1 \leq z)\big)\end{aligned}$$

   The interval constraint for this is: $0 \leq x \wedge x < y_1 - 1.2 \wedge x \leq z - 1$.

3. $$\begin{aligned}&\exists y_1\Big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge \exists y_2\big(a(y_2) \wedge 1 \leq y_2 \leq 2.2\\&\wedge(y_1 + 0.2 < y_2) \wedge \neg\exists y_3(a(y_3) \wedge 1 \leq y_3 \leq 2.2\\&\wedge y_1 < y_3 < y_2) \wedge \exists z(b(z) \wedge y_1 < z)\big)\Big)\end{aligned}$$

   The interval constraint for this is: $y_1 - 1 \leq x \wedge x < y_2 - 1.2 \wedge x \leq z - 1$.

4. $$\begin{aligned}&\exists y_1\big(a(y_1) \wedge 1 \leq y_1 \leq 2.2 \wedge y_1 < 2\\&\wedge\neg\exists y_2(a(y_2) \wedge 1 \leq y_2 \leq 2.2 \wedge y_1 < y_2)\\&\wedge\exists z(b(z) \wedge y_1 < z)\big)\end{aligned}$$

   The interval constraint for this is: $y_1 - 1 \leq x \wedge x \leq 1 \wedge x \leq z - 1$.

We can extend this technique for any number of conjuncts by pushing x from conjuncts one by one, while identifying the corresponding interval constraints simultaneously.

In Lemma 1 we use induction to formalize these ideas and show that it is possible to eliminate passive variables from any formula to derive an equivalent disjunction of active formulae and corresponding interval constraints.

We recall that the *quantifier depth* of an FO formula φ is the maximum number of quantifier nodes along any path from root leaf in the formula tree for φ. We say a quantified variable x in a FO^c formula φ in normal form, is *positively quantified* if in the formula tree of φ, the $\exists x$ node is not present in the subtree rooted at a $\neg\exists y$ node. For example, in the formula $\exists x(\pi(x) \wedge \neg\exists y(\exists z(a(z) \wedge x < z < y)))$, only x is positively quantified and y, z are not positively quantified variables as they are under the scope of $\neg\exists y$. When a formula is satisfied, all positively quantified variables assume some value. For example, if formula (9) is satisfied, then we can say what are the values taken by positively quantified variables y_1, y_2 in the given model, where as we cannot say anything about y_3 which is not positively quantified.

Finally, we introduce a new operator $\overline{\wedge}$ which is a slight modification of $\wedge$ operator. Consider the following formula $\alpha = \varphi \wedge \psi$. Now assume we have an interval constraint $\pi(x)$ which relates the positively quantified variables of φ and ψ. If we want to force α to evaluate to true only when $\pi(x)$ is satisfied, we write it as $\alpha' = \varphi \,\overline{\wedge}\, \psi \,\overline{\wedge}\, \pi(x)$ and this expression will rearrange the brackets of φ and ψ such that all the positively quantified variables of those will be accessible to ψ. Consider this example formula

$$\exists y(a(y) \wedge 1 \leq y) \,\overline{\wedge}\, \exists z(a(z) \wedge z = 2)) \,\overline{\wedge}\, \exists x(y \leq x \leq z)$$

If we replace $\overline{\wedge}$ by $\wedge$ this formula becomes invalid, since y, z are outside the scope of x, but $\overline{\wedge}$ actually pulls y, z into the context of x. Above formula is equivalent to

$$\exists y\Big(a(y) \wedge 1 \leq y \wedge \exists z\big(a(z) \wedge z = 2 \wedge \exists x(y \leq x \leq z)\big)\Big).$$

Formally this operator is defined inductively on the structure of φ as follows:

$$\begin{aligned}a(x) \,\overline{\wedge}\, \psi &= a(x) \wedge \psi\\ g \,\overline{\wedge}\, \psi &= g \wedge \psi\\ \neg\varphi' \,\overline{\wedge}\, \psi &= \neg\varphi' \wedge \psi\\ \exists x(\varphi') \,\overline{\wedge}\, \psi &= \exists x(\varphi' \,\overline{\wedge}\, \psi)\\ (\varphi_1 \wedge \varphi_2) \,\overline{\wedge}\, \psi &= (\varphi_1 \,\overline{\wedge}\, \varphi_2) \,\overline{\wedge}\, \psi\\ (\varphi_1 \vee \varphi_2) \,\overline{\wedge}\, \psi &= (\varphi_1 \,\overline{\wedge}\, \psi) \vee (\varphi_2 \,\overline{\wedge}\, \psi)\end{aligned}$$

Lemma 1 *Let ψ be a formula of the form $\exists x(\pi(x) \wedge \alpha)$ where $\pi(x)$ is an interval constraint for x and α is a conjunction of active formulae in $\exists$-normal form or in negated $\exists$-normal form. Then*

- *we can eliminate x from ψ and give an equivalent formula of the form $\bigvee_i \mu_i$ where each μ_i is in active normal form and does not contain x.*

- *Further, for each μ_i we have an interval constraint $\pi_i(x)$ containing the positively quantified variables $y^i_1, y^i_2, \ldots, y^i_{k_i}$ of μ_i such that: $\sigma, \mathbb{I}[t/x] \models \pi(x) \wedge \alpha$ iff for some i, $\sigma, \mathbb{I} \models \mu_i$ with values $v_1, v_2, \ldots, v_{k_i}$ for $y^i_1, y^i_2, \ldots, y^i_{k_i}$ such that $\mathbb{I}[t/x, v_1/y^i_1, \ldots, v_{k_i}/y^i_{k_i}] \models \pi_i$.*

Proof We proceed by induction on the quantifier depth d of α.

Base case: For the base case of the induction we have $d = 0$. Thus ψ is of the form $\exists x \pi(x)$. The equivalent formula is $fm_x(\pi)$ and interval constraint is $\pi(x)$ itself.

Induction step: Let α have quantifier depth $d+1$. So α is of the form $\alpha_1 \wedge \cdots \wedge \alpha_n$ where each α_i is an active formula in $\exists$-normal form or negated $\exists$-normal form, of depth at most $d+1$. We now proceed by induction on number of conjuncts n of α.

Base case with single conjunct: First we consider the base case where $n = 1$. Here ψ is either of the form $\exists x(\pi(x) \wedge \beta)$ or $\exists x(\pi(x) \wedge \neg\beta)$ where $\beta = \exists y(a(y) \wedge \delta \wedge \gamma)$. Let us write δ as $\delta'(x, y) \wedge \delta''(y)$, where $\delta'(x, y)$ are the constraints in δ containing x. We treat these two cases below.

Single positive conjunct case: In this case, $\psi = \exists x(\pi(x) \wedge \beta) = \exists x(\pi(x) \wedge \exists y(a(y) \wedge \delta'(x, y) \wedge \delta''(y) \wedge \gamma))$. This formula is equivalent to

$$\exists y(a(y) \wedge \delta''(y) \wedge \exists x(\pi(x) \wedge \delta'(x, y) \wedge \gamma)).$$

Since γ has quantifier depth d or less, by induction hypothesis $\exists x(\pi(x) \wedge \delta'(x, y) \wedge \gamma)$ can be translated to $\vee_i \mu_i$ with interval constraints π_i. The final formula is then $\exists y(a(y) \wedge \delta''(y) \wedge (\vee_i \mu_i))$, which is equivalent to

$$\vee_i(\exists y(a(y) \wedge \delta''(y) \wedge \mu_i)).$$

Let $\mu'_i = \exists y(a(y) \wedge \delta''(y) \wedge \mu_i)$. We prove that the corresponding interval constraint for x for each i is $\pi_i(x)$ itself.

($\rightarrow$): Suppose $\sigma, \mathbb{I}[t/x] \models \pi \wedge \beta$ and the value of positively quantified variable y be t_1. This implies that $\sigma, \mathbb{I}[t/x, t_1/y] \models \pi \wedge \delta'(x, y) \wedge \gamma$. By induction hypothesis, for some i, $\sigma, \mathbb{I}[t_1/y] \models \mu'_i$ and $\mathbb{I}[t/x, t_1/y] \models \pi_i$. At $y = t_1$, $a(t_1) \wedge \delta''(t_1)$ is satisfied, which implies, μ'_i is also satisfied at $y = t_1$.

($\leftarrow$): Suppose $\sigma, \mathbb{I} \models \mu'_i$ with positively quantified variable y taking value t_1, then, $\sigma, \mathbb{I}[t_1/y] \models \mu_i$. By induction hypothesis, for every t such that , $\mathbb{I}[t/x, t_1/y] \models \pi_i$ $\sigma, \mathbb{I}[t/x, t_1/y] \models \exists x(\pi(x) \wedge \delta'(x, y) \wedge \gamma)$. Since at $y = t_1$, $a(t_1) \wedge \delta''(t_1)$ is satisfied, $\sigma, \mathbb{I}[t/x] \models \pi \wedge \beta$.

Single negative conjunct case: In this case, $\psi = \exists x(\pi(x) \wedge \neg\beta) = \exists x(\pi(x) \wedge \neg\exists y(a(y) \wedge \delta'(x, y) \wedge \delta''(y) \wedge \gamma))$. Let $\psi' = \exists x(\pi(x) \wedge \beta))$ which falls into the previous case and we can give an equivalent formula $\vee_i \mu'_i$ with corresponding interval constraints $\pi_i(x)$.

For this formula to hold true for any σ in an interpretation for all the free variables, we must be able to identify the gaps in the interval defined by $\pi(x)$ where ψ' is not satisfied. If ψ' is not at all satisfied in the given model, then whole of the interval corresponding to $\pi(x)$ satisfy ψ. If not, we identify the gaps between the intervals $\pi_i(x)$ where ψ' is satisfied. As discussed, there can be 3 different gaps, one before the beginning of the first interval satisfying ψ', two the gap from the right boundary of one interval to the left boundary of another interval where those two are consecutive intervals satisfying ψ' and the third one is the interval gap from the last interval satisfying ψ' to the end of interval defined by π.

We use the interval constraints which define the ordering of the intervals (explained in the preliminaries section) to identify the first, consecutive and the last intervals $\pi_i(x)$ in $\pi(x)$. For simplicity, when we use these interval constraints in the equivalent expressions (12),(13), (14),(15) we assume that x is eliminated from these constraints using the Fourier-Motzkin elimination method.

Let μ'_i be the formula obtained by replicating formula μ_i, but renaming every positively quantified variable y of μ_i by y'. If $y_1, \ldots y_n$ are the positively quantified variables of μ_i, $\mu'_i = \mu_i[y'_1/y_1, \ldots y'_n/y_n]$. If π_i is the constraint of μ_i, then π'_i is the constraint of μ'_i, obtained by renaming positively quantified variables in π_i.

We prove that ψ is equivalent to the disjunction of the following formulae, with the corresponding interval constraints:

1. $$fm_x(\pi) \bigwedge_i \neg\mu_i. \tag{12}$$

   The interval constraint is $\pi(x)$.

2. $$\begin{aligned} &\bigvee_i \Big(\mu_i \mathbin{\bar\wedge} fm_x(l(\pi) \wedge \neg l(\pi_i)) \\ &\mathbin{\bar\wedge} \bigwedge_j \neg\Big(\mu'_j \mathbin{\bar\wedge} fm_x(l(\pi_j) \wedge \neg l(\pi_i))\Big)\Big). \end{aligned} \tag{13}$$

   The interval constraint for each formula(i) is: $l(\pi) \wedge \neg l(\pi_i)$.

3. $$\begin{aligned} &\bigvee_i \bigvee_j \Big(\mu_i \mathbin{\bar\wedge} \mu_j \mathbin{\bar\wedge} fm_x(\neg r(\pi_i) \wedge \neg l(\pi_j)) \mathbin{\bar\wedge} \\ &\bigwedge_k \neg\Big(\mu'_k \mathbin{\bar\wedge} fm_x(\pi_k \wedge \neg r(\pi_i) \wedge \neg l(\pi_j))\Big)\Big). \end{aligned} \tag{14}$$

   The interval constraint for x for the (i, j)-th disjunct is: $\neg r(\pi_i) \wedge \neg l(\pi_j)$.

4.

$$\bigvee_i \Big(\mu_i \mathbin{\overline{\wedge}} fm_x(\neg r(\pi_i) \wedge r(\pi)) \\ \mathbin{\overline{\wedge}} \bigwedge_j \neg\Big(\mu'_j \mathbin{\overline{\wedge}} fm_x(\neg r(\pi_i) \wedge r(\pi_j))\Big)\Big). \tag{15}$$

The interval constraint for each formula(i) is: $\neg r(\pi_i) \wedge r(\pi)$.

($\rightarrow$) Suppose ψ is satisfied at $x = t$ for some σ and for some interpretation $\mathbb{I}$, then ψ' is not satisfied at $x = t$. Consider the intervals $\pi_i(x)$ and $\pi_j(x)$ which satisfy ψ' where $\pi_i(x)$ ends before $x = t$ and no other interval ends after $\pi_i(x)$ and before t. Similarly $\pi_j(x)$ starts after $x = t$ and no other interval starts after t and before $\pi_j(x)$. Now every point in the interval $\neg r(\pi_i) \wedge \neg l(\pi_j)$ including t does not satisfy ψ' and hence satisfy ψ. This implies, (14) is true and $\mathbb{I}[t/x, I] \models \neg r(\pi_i) \wedge \neg l(\pi_j)$, where I represents the interpretation for the positively quantified variables of μ_i and μ_j.

But in the interval π we may not be able to find out $\pi_i(x)$ or $\pi_j(x)$ or both. This is handled by formulae (13), (15), (12) and their corresponding interval constraints.

($\leftarrow$)Let $x = t$ satisfy the equation (14) for some σ and for some interpretation $\mathbb{I}$. Then, $t \in \neg r(\pi_i) \wedge \neg l(\pi_j)$ and in this interval, μ_i is false for all i. This implies ψ' is false at $x = t$ and ψ is true at that point. Similar argument holds true when any of the (12), (13) and (15) holds true at $x = t$.

Multiple conjuncts: Consider formulae of the form $\psi = \exists x(\pi \wedge \alpha_1 \wedge \alpha_2)$. By the second induction hypothesis we know that $\exists x(\pi \wedge \alpha_1)$ can be written as $\vee_i(\mu_i)$ with corresponding interval constraints $\pi_i(x)$. We can prove that $\exists x(\pi \wedge \alpha_1 \wedge \alpha_2) = \bigvee_i \Big(\mu_i \mathbin{\overline{\wedge}} \exists x(\pi_i \wedge \alpha_2)\Big)$.

($\rightarrow$) If $x = t$ satisfies L.H.S. for any σ and for any interpretation $\mathbb{I}$, then it should satisfy $\pi \wedge \alpha_1$, which implies μ_i is satisfied and $\sigma, \mathbb{I}[t/x] \models \pi_i$ for some i. Since t must also satisfy α_2, t will satisfy R.H.S.

($\leftarrow$) If $x = t$ satisfies R.H.S., then $\sigma, \mathbb{I}[t/x] \models \pi_i$, it will satisfy $\pi \wedge \alpha_1$. But t also satisfies α_2 which implies it will satisfy L.H.S.

Consider the formulae of the form $\bigvee_i \Big(\mu_i \mathbin{\overline{\wedge}} \exists x(\pi_i \wedge \alpha_2)\Big)$. From the single conjunct case, we know that it is possible to write the formulae of form $\exists x(\pi_i \wedge \alpha_2)$ as $\bigvee_j(\nu_{i,j})$ with corresponding interval constraints $\delta_{i,j}(x)$. By expanding over $\vee$s, we get

$\psi = \bigvee_i \bigvee_j (\mu_i \mathbin{\overline{\wedge}} \nu_{i,j})$ with a corresponding interval constraint $\delta_{i,j}(x)$.

It is possible to extend this to any number of disjuncts α_i in ψ. □

We can now prove our main theorem.

Theorem 2 *The logics* $\mathrm{FO}^c(<,+)$ *and* $\mathrm{FO}^{pw}(<,+)$ *are expressively equivalent.*

Proof As noted in the beginning of this section, it is sufficient to show that we can express every $\mathrm{FO}^c(<,+)$ formula φ by an equivalent active $\mathrm{FO}^c(<,+)$ formula. Any FO^c formula φ can be written as a boolean combination of $\exists$-normal form formulae. Since translation is intact across the boolean operations, it is sufficient if we can eliminate all the passive variables from formulae in $\exists$-normal form. Consider the formula tree of a formula in $\exists$-normal form. Identify those nodes which are passive and having only active nodes as ancestors. It is easy to see that, if we can eliminate passively quantified variables from each of these trees rooted at these nodes, translation is complete. Now consider any specific node(N), which is of the form $\psi = \exists x(\pi(x) \wedge \bigwedge_{i \le t} \nu_i)$.

We argue that it is possible to convert any passively quantified formula of the form ψ, into an equivalent formula with finite disjunction of active formulae. We prove this based on induction on number of passive variables n in the tree rooted under N.

As a base case, if $n = 1$, the passive variable must be x itself and by Lemma 1, we can convert this formula into an equivalent formula which is a finite disjunction of active formulae. By induction, it is possible to convert any formula with $n < r$ passive variables into an equivalent formula with finite disjunction of active formulae. Consider ψ with $n = r$ number of passive variables. Since x itself is a passive variable and x is a free variable in each of ν_i, they will have utmost $r - 1$ passive variables. But by induction each ν_i can be expressed as finite disjunction of μ_j's , which are active formulae. Then we can normalise the tree below $\exists x$ node and write the expression,

$$\bigvee_{i \le t, j \le v} \exists x\Big(\pi(x) \wedge \mu'_{i,j}\Big) \tag{16}$$

where each of the $\mu'_{i,j}$ is in $\exists$-normal form. Since x is the only passive variable in all of the above, using Lemma 1, we can translate (16), to disjunction of active formulae. □

6 Complexity of Translation

In the second step of the normalization procedure size of the formula can increase exponentially. The translation of FO^c sentence in normal form to a FO^{pw} sentence can take exponential space. In the FM method, eliminating a quantified variable in general leads to a quadratic increase in the number of constraints, i.e. if there are m constraints prior to the elimination, there could be $O(m^2)$ constraints after elimination. Thus elimination of k quantifiers could increase the size of constraint set to $O(m^{2^k})$. The translation of formulae of the form of $\psi = \exists x(\pi(x) \wedge \neg\beta)$ introduces four formulae where each one is proportional to the length of the input formula. This also contributes to the exponential growth of the translated formula.

7 Equivalence of semantics for TPTL_S

We recall that the formulae of TPTL_S [2] over the alphabet Σ, are defined by the syntax

$$\theta ::= a \mid g \mid \neg\theta \mid (\theta \vee \theta) \mid (\theta U \theta) \mid (\theta S \theta) \mid x.\theta,$$

where $a \in \Sigma$, x is a variable in Var, and g is a simple constraint. We first define the *pointwise* semantics for TPTL_S. Let θ be a TPTL_S formula. Let $\sigma = (\alpha, \tau)$ be a timed word over Σ, let $i \in \mathbb{N}$, and let $\mathbb{I}$ be an interpretation for variables. Then the satisfaction relation $\sigma, i, \mathbb{I} \models_{pw} \theta$ is defined (omitting boolean operators) as:

$\sigma, i, \mathbb{I} \models_{pw} a$	iff	$\alpha(i) = a$
$\sigma, i, \mathbb{I} \models_{pw} g$	iff	$\mathbb{I} \models g$
$\sigma, i, \mathbb{I} \models_{pw} \neg\theta$	iff	$\sigma, i, \mathbb{I} \not\models_{pw} \theta$
$\sigma, i, \mathbb{I} \models_{pw} \theta \vee \eta$	iff	$\sigma, i, \mathbb{I} \models_{pw} \theta$ or $\sigma, i, \mathbb{I} \models_{pw} \eta$
$\sigma, i, \mathbb{I} \models_{pw} \theta U \eta$	iff	$\exists k : k > i$ s.t. $\sigma, k, \mathbb{I} \models_{pw} \eta$ and $\forall j : i < j < k : \sigma, j, \mathbb{I} \models_{pw} \theta$
$\sigma, i, \mathbb{I} \models_{pw} \theta S \eta$	iff	$\exists k : 0 \leq k < i$ s.t. $\sigma, k, \mathbb{I} \models_{pw} \eta$ and $\forall j : k < j < i, \sigma, j, \mathbb{I} \models_{pw} \theta$
$\sigma, i, \mathbb{I} \models_{pw} x.\theta$	iff	$\sigma, i, \mathbb{I}[\tau(i)/x] \models_{pw} \theta$

For the *continuous* semantics for TPTL_S, the satisfaction relation $\sigma, t, \mathbb{I} \models_c \theta$ (where $t \in \mathbb{R}_{\geq 0}$) defined as:

$\sigma, t, \mathbb{I} \models_c a$	iff	$\exists i : \alpha(i) = a$ and $\tau(i) = t$
$\sigma, t, \mathbb{I} \models_c g$	iff	$\mathbb{I} \models g$
$\sigma, t, \mathbb{I} \models_c \neg\theta$	iff	$\sigma, t, \mathbb{I} \not\models_c \theta$
$\sigma, t, \mathbb{I} \models_c \theta \vee \eta$	iff	$\sigma, t, \mathbb{I} \models_c \theta$ or $\sigma, t, \mathbb{I} \models_c \eta$
$\sigma, t, \mathbb{I} \models_c \theta U \eta$	iff	$\exists t' : t' > t$ s.t. $\sigma, t', \mathbb{I} \models_c \eta$ and $\forall t'' : t < t'' < t', \sigma, t'', \mathbb{I} \models_c \theta$
$\sigma, t, \mathbb{I} \models_c \theta S \eta$	iff	$\exists t' : 0 \leq t' < t$ s.t. $\sigma, t', \mathbb{I} \models_c \eta$ and $\forall t'' : t' < t'' < t, \sigma, t'', \mathbb{I} \models_c \theta$
$\sigma, t, \mathbb{I} \models_c x.\theta$	iff	$\sigma, t, \mathbb{I}[t/x] \models_c \theta$

We use the standard syntactic abbreviations of $\Diamond, \overline{\Diamond}, \Box$ and $\overline{\Box}$ for temporal logic, defined in a way that they are "reflexive". Thus, we define $\Diamond\theta = \theta \vee (\top U \theta)$, $\overline{\Diamond}\theta = \theta \vee (\top S \theta)$, $\Box\theta = \neg\Diamond\neg\theta$, $\overline{\Box}\theta = \neg\overline{\Diamond}\neg\theta$. We note that in the logic TPTL_S, it is possible to express both U and S operators using $\Diamond$ and $\overline{\Diamond}$ operators in both the continuous and pointwise semantics.

$$\begin{aligned} \theta U \eta &\equiv x.\Diamond y.(\eta \wedge x < y \wedge \overline{\Box} z.(x < z \wedge z < y \Rightarrow \theta)) \\ \theta S \eta &\equiv x.\overline{\Diamond} y.(\eta \wedge y < x \wedge \Box z.(y < z \wedge z < x \Rightarrow \theta)). \end{aligned}$$

To see that TPTL_S and $\mathrm{FO}(<,+)$ are expressively equivalent in the pointwise as well as continuous semantics, we first note that going from TPTL_S to $\mathrm{FO}(<,+)$ is a standard translation. In the other direction, we translate an $\mathrm{FO}(<,+)$ sentence φ, to an equivalent closed TPTL_S formula $\textit{fo-tptl}(\varphi)$ as follows. We first transform φ into its normal form as given in Theorem(1). In the translation *fo-tptl* we essentially translate the $\exists$ subformulae via the rule:

$$\begin{aligned} \textit{fo-tptl}(\exists x(a(x) \wedge \pi(x) \wedge \nu)) = {} & \Diamond x.(a \wedge \pi(x) \wedge \textit{fo-tptl}(\nu)) \vee \\ & \overline{\Diamond} x.(a \wedge \pi(x) \wedge \textit{fo-tptl}(\nu)). \end{aligned}$$

Hence, using Theorem 2, we have

Theorem 3 *The pointwise and continuous versions of the logic* TPTL_S *are expressively equivalent.*

8 Discussion

We have shown in this paper that the pointwise and continuous semantics of the logic $\mathrm{FO}(<,+)$ coincide. The main technical contribution is a way to translate formulae in $\mathrm{FO}^c(<,+)$ to equivalent ones in $\mathrm{FO}^{pw}(<,+)$. We note that our results also hold for the logic with general linear constraints of the form $a_1x_1 + a_2x_2 + \cdots + a_nx_n \sim c$ (which we could denote by $\mathrm{FO}(<,+)$). The only reason we chose to work with simple constraints was to be able to deduce the result for TPTL_S which is defined with such constraints.

In future work we need to investigate further the complexity of our translation procedure in terms of both upper and lower bounds. Also it would be interesting to answer the question of relative expressiveness of the pointwise and continuous semantics for the logic TPTL without the since operator.

References

[1] R. Alur, T. Feder, and T. A. Henzinger. The benefits of relaxing punctuality. *J. ACM*, 43(1):116–146, 1996.

[2] R. Alur and T. A. Henzinger. A really temporal logic. *J. ACM*, 41(1):181–204, 1994.

[3] P. Bouyer, F. Chevalier, and N. Markey. On the expressiveness of tptl and mtl. In R. Ramanujam and S. Sen, editors, *FSTTCS*, volume 3821 of *Lecture Notes in Computer Science*, pages 432–443. Springer, 2005.

[4] D. D'Souza and P. Prabhakar. On the expressiveness of mtl in the pointwise and continuous semantics. *STTT*, 9(1):1–4, 2007.

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

[6] J. Ouaknine and J. Worrell. On the decidability of metric temporal logic. In *LICS*, pages 188–197. IEEE Computer Society, 2005.

[7] P. Prabhakar and D. D'Souza. On the expressiveness of mtl with past operators. In E. Asarin and P. Bouyer, editors, *FORMATS*, volume 4202 of *Lecture Notes in Computer Science*, pages 322–336. Springer, 2006.

[8] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, Inc., New York, NY, USA, 1986.

Appendix

Lemma 2 *Step 2 of normalization procedure terminates.*

Proof Consider the formula tree of φ. Color a, $\neg a$ and g nodes green. Color $\exists$ and $\neg\exists$ nodes red if the subtree rooted at these nodes contains an $\vee$ node, otherwise color it green.

Consider a red node $\exists\alpha$ or $\neg\exists\alpha$ which does not contain any other red nodes in its subtree. The subtree of α is a combination of conjunctions and disjunctions of green nodes. We treat each green node as an atomic node and rewrite it in disjunctive normal form (DNF). After this step, node α will be of the form $\vee_i\alpha_i$ where each α_i does not contain any $\vee$ operator. We can pull these $\vee$ operators using the equivalence $\exists x(\alpha_1 \vee \alpha_2) \equiv (\exists x\alpha_1) \vee (\exists x\alpha_2)$ and $\neg\exists x(\alpha_1 \vee \alpha_2) \equiv (\neg\exists x(\alpha_1)) \wedge (\neg\exists x(\alpha_2))$. The node will look like $\vee_i\exists\alpha_i$ or $\wedge_i\neg\exists\alpha_i$ where each $\exists\alpha_i$ or $\neg\exists\alpha_i$ is now colored green.

We repeat the above procedure, till there are no red nodes in the formula tree. In each step, we reduce one red node and do not introduce any new red nodes. This implies the procedure terminates. Finally, as desired all the nodes will be green and all $\vee$ operators are pulled up to the top of the tree. □

For a closed formula θ in TPTL_S^c we show how to give a formula $\mathit{tptl\text{-}fo}(\theta)$ in $\mathrm{FO}^c(<,+)$, which has a single free variable z, such that for any timed word σ, $\sigma, t \models_c \theta$ if and only if $\sigma, [t/z] \models_c \mathit{tptl\text{-}fo}(\theta)$. The translation $\mathit{tptl\text{-}fo}$ is defined inductively on the structure of θ as follows:

$$
\begin{array}{lcl}
\mathit{tptl\text{-}fo}(a) & = & a(z) \\
\mathit{tptl\text{-}fo}(g) & = & g \\
\mathit{tptl\text{-}fo}(\neg\theta') & = & \neg\mathit{tptl\text{-}fo}(\theta') \\
\mathit{tptl\text{-}fo}(\theta_1 \wedge \theta_2)) & = & \mathit{tptl\text{-}fo}(\theta_1) \wedge \mathit{tptl\text{-}fo}(\theta_2) \\
\mathit{tptl\text{-}fo}(\Diamond\theta') & = & \exists x(x \geq z \wedge \mathit{tptl\text{-}fo}(\theta'[x/z])) \\
\mathit{tptl\text{-}fo}(\overline{\Diamond}\theta') & = & \exists x(x \leq z \wedge \mathit{tptl\text{-}fo}(\theta'[x/z])) \\
\mathit{tptl\text{-}fo}(x.\theta') & = & (\mathit{tptl\text{-}fo}(\theta'))[z/x]
\end{array}
$$

Now we can translate a closed formula θ in TPTL_S^c to the $\mathrm{FO}^c(<,+)$ sentence $\varphi = \exists z(z = 0 \wedge \mathit{tptl\text{-}fo}(\theta))$, so that we have $L^c(\theta) = L^c(\varphi)$.