Delft University of Technology

From Non-punctuality to Non-adjacency
A Quest for Decidability of Timed Temporal Logics with Quantifiers

Krishna, Shankara Narayanan; Madnani, Khushraj; Mazo, Manuel; Pandya, Paritosh

DOI
10.1145/3571749

Publication date
2023

Document Version
Final published version

Published in
Formal Aspects of Computing

Citation (APA)
Krishna, S. N., Madnani, K., Mazo, M., & Pandya, P. (2023). From Non-punctuality to Non-adjacency: A Quest for Decidability of Timed Temporal Logics with Quantifiers. *Formal Aspects of Computing*, *35*(2), Article 9. https://doi.org/10.1145/3571749

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

From Non-punctuality to Non-adjacency: A Quest for Decidability of Timed Temporal Logics with Quantifiers

SHANKARA NARAYANAN KRISHNA, Indian Institute of Technology Bombay, India
KHUSHRAJ MADNANI, Max Planck Institute for Software Systems, Germany
MANUEL MAZO JR., Delft University of Technology, The Netherlands
PARITOSH PANDYA, Indian Institute of Technology Bombay, India

9

Metric Temporal Logic (MTL) and Timed Propositional Temporal Logic (TPTL) are prominent real-time extensions of Linear Temporal Logic (LTL). In general, the satisfiability checking problem for these extensions is undecidable when both the future (Until, U) and the past (Since, S) modalities are used (denoted by MTL[U,S] and TPTL[U,S]). In a classical result, the satisfiability checking for Metric Interval Temporal Logic (MITL[U,S]), a non-punctual fragment of MTL[U,S], is shown to be decidable with EXPSPACE complete complexity. A straightforward adoption of non-punctuality does not recover decidability in the case of TPTL[U,S]. Hence, we propose a more refined notion called *non-adjacency* for TPTL[U,S] and focus on its 1-variable fragment, 1-TPTL[U,S]. We show that non-adjacent 1-TPTL[U,S] is strictly more expressive than MITL. As one of our main results, we show that the satisfiability checking problem for non-adjacent 1-TPTL[U,S] is decidable with EXPSPACE complete complexity. Our decidability proof relies on a novel technique of anchored interval word abstraction and its reduction to a non-adjacent version of the newly proposed logic called PnEMTL. We further propose an extension of MSO [<] (Monadic Second Order Logic of Orders) with Guarded Metric Quantifiers (GQMSO) and show that it characterizes the expressiveness of PnEMTL. That apart, we introduce the notion of non-adjacency in the context of GQMSO (NA-GQMSO), which is a syntactic generalization of logic Q2MLO due to Hirshfeld and Rabinovich and show the decidability of satisfiability checking for NA-GQMSO.

CCS Concepts: • **Theory of computation** → **Modal and temporal logics**; **Timed and hybrid models**; **Formal languages and automata theory;**

Additional Key Words and Phrases: Real-time logics, Metric Temporal Logic, Timed Propositional Temporal Logic, satisfiability checking, non-punctuality, decidability, expressiveness

ACM Reference format:
Shankara Narayanan Krishna, Khushraj Madnani, Manuel Mazo Jr., and Paritosh Pandya. 2023. From Non-punctuality to Non-adjacency: A Quest for Decidability of Timed Temporal Logics with Quantifiers. *Form. Asp. Comput.* 35, 2, Article 9 (June 2023), 50 pages.
https://doi.org/10.1145/3571749

This work is partially supported by the European Research Council through the SENTIENT project (ERC-2017-STG #755953).

Authors' addresses: S. N. Krishna and P. Pandya, Indian Institute of Technology Bombay, Mumbai, Maharashtra, India, 400076; emails: krishnas@cse.iitb.ac.in, pandya58@iitb.ac.in; K. Madnani, Max Planck Institute for Software Systems, Kaiserslautern, Germany, 67663; email: kmadnani@mpi-sws.org; M. Mazo Jr., Delft University of Technology, Delft, South Holland, The Netherlands, 2628CD; email: M.Mazo@tudelft.nl.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
© 2023 Copyright held by the owner/author(s).

0934-5043/2023/06-ART9 $15.00
https://doi.org/10.1145/3571749

1 INTRODUCTION

Metric Temporal Logic (MTL) and **Timed Propositional Temporal Logic (TPTL)** are natural extensions of **Linear Temporal Logic (LTL)** for specifying real-time properties [5]. MTL extends the Until (U) and Since (S) modalities of LTL by associating a timing interval with these. $a\mathsf{U}_I b$ describes behaviors modelled as timed words consisting of a sequence of a's followed by a b, which occurs at a time within (relative) interval I. However, TPTL uses freeze quantification to store the current timestamp. A freeze quantifier (also called as Half Order Quantifiers [3]) with clock variable x has the form $x.\varphi$. When it is evaluated at a point i on a timed word, the timestamp τ_i at i is frozen or registered in x, and the formula φ is evaluated using this value for x. Variable x is used in φ in a constraint of the form $T - x \in I$; this constraint, when evaluated at a point j, checks if $\tau_j - \tau_i \in I$, where τ_j is the timestamp at point j.[1] For example, the formula $\mathsf{F}x.(a \wedge \mathsf{F}(b \wedge T - x \in [1,2] \wedge \mathsf{F}(c \wedge T - x \in [1,2])))$ asserts that there is a point in the future where a holds and in its future within interval $[1,2]$, b and c occur, and b occurs before c. This property is not expressible in MTL[U, S] [8, 39]. Moreover, every property in MTL[U, S] can be expressed in 1 variable fragment of TPTL (1-TPTL[U, S]). Thus, 1-TPTL[U, S] is strictly more expressive than MTL[U, S]. Unfortunately, both the logics have an undecidable satisfiability checking problem, making automated analysis of these logics difficult (in full generality existence of a sound and complete algorithm is impossible for such problems). It is possible to restrict certain parameters of the behaviors and get terminating algorithms. But that would require prior information about some parameters of the behaviors, which may not be always accessible. Moreover, the complexity of the algorithm often depends on the value of these parameters. For example, if we restrict the models to be k-bounded variable, i.e., models where the number of events within any unit time interval is bounded by k,[2] then the satisfiability checking becomes decidable for these logics [19] but the complexity of this problem depends on k. Moreover, this would require access to this bound k, which is not the case, in general. Exploring natural decidable variants of these logics has been an active area of research since their advent [4, 23, 25, 26, 42, 43, 46]. One line of work restricted the logic to contain future only modality MTL[U] and 1-TPTL[U]. Both these logics have been shown to have decidable satisfiability over finite timed words, under a pointwise interpretation [22, 37].[3] The complexity, however, is non-primitive recursive. Moreover, these problems become undecidable over infinite timed words. Obtaining an expressive fragment with elementary complexity has been a challenging problem. One of the most celebrated such logics is the **Metric Interval Temporal Logic (MITL[U, S])** [1], a subclass of MTL[U, S] where the timing intervals is restricted to be non-punctual, i.e., non-singular (intervals of the form $\langle x, y\rangle$ where $x < y$). The satisfiability checking for MITL formulae is decidable with EXPSPACE complete complexity [1]. While non-punctuality helps to recover the decidability of MTL[U, S], it does not help in TPTL[U, S]. The freeze quantifiers of TPTL enable us to trivially express punctual timing constraints using only the non-punctual intervals: For instance, the 1-TPTL formula $x.(a\mathsf{U}(a \wedge T - x \in [1,\infty) \wedge T - x \in [0,1]))$ uses only non-punctual intervals but captures the MTL formula

[1]Here, T is a special variable that stores the timestamp of the present point and x is the clock that was frozen when $x.$ was asserted.

[2]Bounded variability is usually defined on timed signals rather than timed words. But, every timed word can be equivalently represented as timed signals. Moreover, the definition of bounded variability of Reference [19] for timed words boils down to the above-mentioned restriction.

[3]While Reference [37] proves decidability of MTL[U] via reduction to **1-clock Alternating Timed Automata (1-ATA)** followed by proving decidability for emptiness checking problem of 1-ATA over finite models. The generalization of this reduction is provided in Reference [22], where the authors prove a stronger result showing that 1-TPTL[U] with least fix-point operator is expressively equivalent to 1-ATA over finite models.

$a\mathsf{U}_{[1,1]}b$. Thus, a more refined notion of non-punctuality is needed to recover the decidability of 1-TPTL[U, S].[4]

Contributions. With the above observations, to obtain a decidable class of 1-TPTL[U, S] akin to MITL[U, S], we revisit the notion of non-punctuality as it stands currently. As our first contribution, we propose *non-adjacency*, a refined version of non-punctuality. Two intervals I_1 and I_2 are non-adjacent if the supremum of I_1 is not equal to the infimum of I_2. Non-adjacent 1-TPTL[U, S] is the subclass of 1-TPTL[U, S], where every interval used in clock constraints within the same freeze quantifier is non-adjacent to itself and to every other timing interval that appears within the same scope. (W.l.o.g., we consider formulae in negation normal form only.) The non-adjacency restriction disallows punctual timing intervals: Every punctual timing interval is adjacent to itself. It can be shown (Theorem 3.2) that non-adjacent 1-TPTL[U, S], while seemingly very restrictive, is strictly more expressive than MITL and it can also express the counting and the Pnueli modalities [25]. Thus, the logic is of considerable interest in practical real-time specification (see Example 3.1).

Our second contribution is to give a decision procedure for the satisfiability checking of non-adjacent 1-TPTL[U, S]. We do this in two steps. (1) We introduce a logic PnEMTL that combines and generalizes the automata modalities of References [27, 43, 46] and the Pnueli modalities of References [25, 26, 42] and has not been studied before to the best of our knowledge. We show that a formula in non-adjacent 1-TPTL[U, S] can be reduced to an equivalent formula of non-adjacent PnEMTL (Theorem 5.16). (2) We prove that the satisfiability of non-adjacent PnEMTL is decidable with EXPSPACE complete complexity (Theorem 8.1) by reducing it to an equisatisfiable $\mathrm{EMITL}_{0,\infty}$ formulae (subclass of EMTL where the timing constraints are restricted to be of the form $\langle 0, u\rangle$ or $\langle l, \infty\rangle$ where $l, u \in \mathbb{N} \cup \{0\}$).

As our third and final contribution, we show that the logic PnEMTL is expressively equivalent to an extension of **MSO[<] (Monadic Second Order Logic of Orders)** with **Guarded Metric Quantifiers (GQMSO)**. The latter is a versatile and expressive logic, allowing properties of real-time systems to be defined conveniently. The use of Guarded Metric Quantifiers appeared in the pioneering formulations of logics QMLO and Q2MLO (with non-punctual guards) by Hirshfeld and Rabinovich [25] and it was further explored by Hunter (with punctual guards) [28]. We have **generalized** these to an anchored block of guarded quantifiers with arbitrary depth. This provides the required power to obtain expressive completeness. We show this by providing effective reductions from PnEMTL to GQMSO and vice versa. Unfortunately, the full PnEMTL, being a syntactic extension of MTL[U, S], is clearly undecidable. As our final main result, we define the non-adjacency condition, suitably applied to the logic GQMSO. We observe that the effective reductions between GQMSO and PnEMTL preserve non-adjacency. From the previously established EXPSPACE-complete decidability of non-adjacent PnEMTL, it follows that the satisfiability checking for non-adjacent GQMSO is decidable.

The article is organized as follows: Section 2 introduces the models and logics LTL, MTL, and TPTL. Section 3 introduces MTL extended with **Pnueli automata modalities (PnEMTL)** and non-adjacent fragments of $1-$TPTL and PnEMTL. Section 4 introduces a novel notion of anchored interval word abstractions that we use to abstract timed languages. Its theory is central in the reduction of any (non-adjacent) TPTL formula to an equivalent (non-adjacent) PnEMTL formula

[4] Even if we restrict the syntax to disallow Boolean expressions over constraints having a unique solution, it is possible to get undecidability due to the power of freeze quantification. The main power of adjacency comes from the fact that it could express the following kind of properties: a holds at the last/first point within the next/previous unit interval. For example, $x.[\mathsf{F}\{a \wedge x \in (0,1) \wedge \oplus(x \in (1,\infty))\}]$ (symbol $\oplus$ stands for the next operator) specifies a holds at the last point in the next unit interval. This property can then be used to encode runs of any arbitrary 2 counter machines. See Reference [35], Chapter 3, Section 3.4, for more details.

presented in Section 5. Section 6 introduces a new extension of **MSO[<] with Guarded Metric Quantifiers (GQMSO)** and its **non-adjacent fragment (NA-GQMSO)**. As mentioned, this logic is a natural syntactic generalization of QMLO and Q2MLO of Reference [26] and QkMSO of Reference [33]. In Section 7, we show that PnEMTL (and non-adjacent PnEMTL) is equivalent to GQMSO (and non-adjacent GQMSO, respectively) by giving effective reductions in both directions. Finally, Section 8 shows that satisfiability checking for non-adjacent PnEMTL and $1-\mathrm{TPTL}$ is decidable with EXSPACE complete complexity. This, along with the reduction from non-adjacent GQMSO to non-adjacent PnEMTL, implies that the satisfiability checking problem for non-adjacent GQMSO is decidable. Finally, in Section 10, we conclude our article and discuss its place in the existing literature. We finish by proposing a fundamental open question in timed logics.

Discussion and related work. Much of the related work has already been discussed. MITL with counting and Pnueli modalities has been shown to have EXPSPACE-complete satisfiability [41, 42]. Here, we tackle more expressive logics, namely, non-adjacent 1-TPTL[U, S] and non-adjacent PnEMTL. We show that the EXPSPACE-complete satisfiability checking is retained in spite of the additional expressive power. These decidability results are proved by equisatisfiable reductions to logic $\mathrm{EMITL}_{0,\infty}$ of Ho [27]. As argued by Ho, it is quite practicable to extend the existing model checking tools like UPPAAL to logic $\mathrm{EMITL}_{0,\infty}$ and hence to our logics, too.

Addition of regular expression-based modalities to untimed logics like LTL has been found to be quite useful for practical specification; even the IEEE standard temporal logic PSL has this feature [15, 18, 29]. With a similar motivation, there has been considerable recent work on adding regular expression/automata-based modalities to MTL and MITL. Raskin as well as Wilke added automata modalities to MITL as well as an Event-Clock logic *ECL* [43, 46] and showed its satisfiability checking problem to be decidable. Krishna et al. showed that $\mathrm{MTL}[\mathrm{U}, \mathrm{S}_{NP}]$ (where U can use punctual intervals but S is restricted to non-punctual intervals), when extended with counting as well as regular expression modalities preserves decidability of satisfaction [31–33, 35]. Recently, Ferrère in Reference [17] proposed a very neat extension of MITL, called **Metric Interval Dynamic Logic (MIDL)**, where the timing constraints appear within regular expressions as opposed to modalities (LTL[U] extended with a fragment of timed regular (MIRE) expression modality). He showed that satisfiability checking for MIDL is decidable with EXPSPACE complete complexity. Moreover, Ho has investigated a PSPACE-complete fragment $\mathrm{EMITL}_{0,\infty}$ and showed that this fragment is surprisingly as expressive as the full logic EMITL [27]. Our non-adjacent PnEMTL is a novel extension of MITL with modalities that combine the features of EMITL [27, 43, 46] and the Pnueli modalities [25, 26, 42]. In terms of expressiveness, MIDL is also known to be strictly more expressive than EMITL. However, the relation between non-adjacent PnEMTL and MIDL remains open.

In terms of expressive completeness, Hirshfeld and Rabinovich [26] showed that MITL is expressively complete to an extension of FO[<] with metric quantifiers (quantifiers guarded with non-punctual timing constraints) where the subformulae within the scope of this metric quantifier is restricted to have only one free variable. Moreover, its extension, Q2MLO (where the subformulae within the scope of the metric quantifier can have no more than 2 free variables), is expressively equivalent to MITL extended with Pnueli Modalities. Hunter [28] showed that when one allows punctual guards in Q2MLO, one gets the complete first-order logic with distance operator $\mathrm{FO}[<, +1]$ in continuous semantics. Inspired by these logics, Reference [33] proposed its extensions with restricted form of second-order quantification giving Q2MSO or QkMSO and allows punctuality. But these logics were restricted to reason about future time properties only, to preserve the decidability. Our proposed logic GQMSO is a syntactic generalization of all these logics.

2 PRELIMINARIES

Let Σ be a finite set of propositions, and let $\Gamma = 2^{\Sigma} \setminus \emptyset$.[5] A word over Σ is a finite sequence $\sigma = \sigma_1\sigma_2\ldots\sigma_n$, where $\sigma_i \in \Gamma$. A timed word ρ over Σ is a non-empty finite sequence $\rho = (\sigma_1, \tau_1)\ldots(\sigma_n, \tau_n)$ of pairs $(\sigma_i, \tau_i) \in (\Gamma \times \mathbb{R}_{\geq 0})$; where $\tau_1 = 0$ and $\tau_i \leq \tau_j$ for all $1 \leq i \leq j \leq n$ and n is the length of ρ (also denoted by $|\rho|$). The τ_i are called timestamps. For a timed or untimed word ρ, let $dom(\rho) = \{i|1 \leq i \leq |\rho|\}$, and $\sigma[i]$ denotes the symbol at position $i \in dom(\rho)$. The set of timed words over Σ is denoted $T\Sigma^*$. Given a (timed) word ρ and $i \in dom(\rho)$, a pointed (timed) word is the pair ρ, i. Let $\mathcal{I}_{\text{int}}$ ($\mathcal{I}_{\text{nat}}$) be the set of open, half-open, or closed time intervals, such that the end points of these intervals are in $\mathbb{Z} \cup \{-\infty, \infty\}$ ($\mathbb{N} \cup \{0, \infty\}$, respectively).

2.1 Linear Temporal Logic

Formulae of LTL are built over a finite set of propositions Σ using Boolean connectives and temporal modalities (U and S) as follows: $\varphi ::= a \,|\top \mid \varphi \wedge \varphi \mid \neg\varphi \mid \varphi\mathsf{U}\varphi \mid \varphi\mathsf{S}\varphi$, where $a \in \Sigma$. The satisfaction of an LTL formula is evaluated over pointed words. For a word $\sigma = \sigma_1\sigma_2\ldots\sigma_n \in \Sigma^*$ and a point $i \in dom(\sigma)$, the satisfaction of an LTL formula φ at point i in σ is defined, recursively, as follows:

(i) $\sigma, i \models a$ iff $a \in \sigma_i$,
(ii) $\sigma, i \models \top$ iff $i \in dom(\rho)$,
(iii) $\sigma, i \models \neg\varphi$ iff $\sigma, i \nvDash \varphi$
(iv) $\rho, i \models \varphi_1 \wedge \varphi_2$ iff $\sigma, i \models \varphi_1$ and $\sigma, i \models \varphi_2$,
(v) $\rho, i \models \varphi_1 \vee \varphi_2$ iff $\sigma, i \models \varphi_1$ or $\sigma, i \models \varphi_2$,
(vi) $\sigma, i \models \varphi_1\mathsf{U}\varphi_2$ iff $\exists j > i, \sigma, j \models \varphi_2$, and $\sigma, k \models \varphi_1 \;\forall\, i < k < j$,
(vii) $\sigma, i \models \varphi_1\mathsf{S}\varphi_2$ iff $\exists j < i, \sigma, j \models \varphi_2$, and $\sigma, k \models \varphi_1 \;\forall\, j < k < i$.

Derived operators can be defined as follows: $\mathsf{F}\varphi = \top\mathsf{U}\varphi$, and $\mathcal{G}\varphi = \neg\mathsf{F}\neg\varphi$. Symmetrically, $\mathsf{P}\varphi = \top\mathsf{S}\varphi$, and $\mathcal{H}\varphi = \neg\mathsf{P}\neg\varphi$. An LTL formula is said to be in negation normal form if it is constructed out of basic and derived operators above, but where negation appears only in front of propositional letters. It is well known that every LTL formula can be converted to an equivalent formula that is in negation normal form.

2.2 Metric Temporal Logic (MTL)

MTL is a real-time extension of LTL where the modalities (U and S) are guarded with intervals. Formulae of MTL are built from Σ using Boolean connectives and time-constrained versions U_I and S_I of the standard U, S modalities, where $I \in \mathcal{I}_{\text{nat}}$. Intervals of the form $[x, x]$ are called punctual; a non-punctual interval is one that is not punctual. Formulae in MTL are defined as follows: $\varphi ::= a \,|\top \,|\varphi \wedge \varphi \mid \neg\varphi \mid \varphi\mathsf{U}_I\varphi \mid \varphi\mathsf{S}_I\varphi$, where $a \in \Sigma$ and $I \in \mathcal{I}_{\text{nat}}$. For a timed word $\rho = (\sigma_1, \tau_1)(\sigma_2, \tau_2)\ldots(\sigma_n, \tau_n) \in T\Sigma^*$, a position $i \in dom(\rho)$, an MTL formula φ, the satisfaction of φ at a position i of ρ, denoted $\rho, i \models \varphi$, is defined below. We discuss the time-constrained modalities.

- $\rho, i \models \varphi_1\mathsf{U}_I\varphi_2$ iff $\exists j > i, \rho, j \models \varphi_2, \tau_j - \tau_i \in I$, and $\rho, k \models \varphi_1 \;\forall\, i < k < j$.
- $\rho, i \models \varphi_1\mathsf{S}_I\varphi_2$ iff $\exists j < i, \rho, j \models \varphi_2, \tau_i - \tau_j \in I$, and $\rho, k \models \varphi_1 \;\forall\, j < k < i$.

The language of an MTL formula φ is defined as $L(\varphi) = \{\rho|\rho, 1 \models \varphi\}$. Using the above, we obtain some derived formulae: the *constrained eventual* operator $\mathsf{F}_I\varphi \equiv \top\mathsf{U}_I\varphi$ and its dual is $\mathcal{G}_I\varphi \equiv \neg\mathsf{F}_I\neg\varphi$. Similarly $\mathcal{H}_I\varphi \equiv \top\mathsf{S}_I\varphi$. The *next* operator is defined as $\oplus_I\varphi \equiv \bot\mathsf{U}_I\varphi$. The non-strict

[5]We exclude this empty-set for technical reasons. This simplifies definitions related to equisatisfiable modulo oversampled projections [35]. Note that this does not affect the expressiveness of the models, as one can add a special symbol denoting the empty-set.

versions of $\mathsf{F}, \mathcal{G}$ are, respectively, defined as $\mathsf{F}^w\varphi \equiv \varphi \vee \mathsf{F}\varphi$ and $\mathcal{G}^w\varphi \equiv \varphi \wedge \mathcal{G}\varphi$ include the present point. Symmetric non-strict versions for past operators are also allowed. The subclass of MTL obtained by restricting the intervals I in the until and since modalities to **non-punctual intervals** is known as **Metric Interval Temporal** logic and denoted by **MITL**$[\mathsf{U}, \mathsf{S}]$. We say that a formula φ is satisfiable iff $L(\varphi) \neq \emptyset$.

Theorem 2.1. *Satisfiability checking for MTL*$[\mathsf{U}, \mathsf{S}]$ *is undecidable [4]. Satisfiability Checking for MITL*$[\mathsf{U}, \mathsf{S}]$ *is EXPSPACE-complete [1–3].*

2.3 Timed Propositional Temporal Logic (TPTL)

The logic TPTL also extends LTL using freeze quantifiers. Like MTL, TPTL is also evaluated on timed words. Formulae of TPTL are built from Σ using Boolean connectives, modalities U and S of LTL. In addition, TPTL uses a finite set of real valued clock variables $X = \{x_1, \ldots, x_n\}$. Let $\nu : X \to \mathbb{R}_{\geq 0}$ represent a valuation assigning a non-negative real value to each clock variable. The formulae of TPTL are defined as follows: Without loss of generality, we work with TPTL in the negation normal form. $\varphi ::= a \mid \neg a \mid \top \mid \bot \mid x.\varphi \mid T - x \in I \mid x - T \in I \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \mathsf{U} \varphi \mid \varphi \mathsf{S} \varphi \mid \mathcal{G}\varphi \mid \mathcal{H}\varphi$, where $x \in X$, $a \in \Sigma$, $I \in \mathcal{I}_{\mathsf{int}}$. Here, T denotes the timestamp of the point where the formula is being evaluated. $x.\varphi$ is the freeze quantification construct that remembers the timestamp of the current point in variable x and evaluates φ.

For a timed word $\rho = (\sigma_1, \tau_1) \ldots (\sigma_n, \tau_n)$, $i \in dom(\rho)$ and a TPTL formula φ, we define the satisfiability relation, $\rho, i, \nu \models \varphi$ with valuation ν of all the clock variables. We omit the semantics of Boolean, U and S operators as they are similar to those of LTL.

- $\rho, i, \nu \models a$ iff $a \in \sigma_i$, and $\rho, i, \nu \models x.\varphi$ iff $\rho, i, \nu[x \leftarrow \tau_i] \models \varphi$,
- $\rho, i, \nu \models T - x \in I$ iff $\tau_i - \nu(x) \in I$, and $\rho, i, \nu \models x - T \in I$ iff $\nu(x) - \tau_i \in I$,
- $\rho, i, \nu \models \mathcal{G}\varphi$ iff $\forall j > i$, $\rho, j, \nu \models \varphi$, and
- $\rho, i, \nu \models \mathcal{H}\varphi$ iff $\forall j < i$, $\rho, j, \nu \models \varphi$.

Let $\overline{0} = (0, 0, \ldots, 0)$ represent the initial valuation of all clock variables. For a timed word ρ and $i \in dom(\rho)$, we say that ρ, i satisfies φ denoted $\rho, i \models \varphi$ iff $\rho, i, \overline{0} \models \varphi$. The language of φ, $L(\varphi) = \{\rho | \rho, 1 \models \varphi\}$. The Pointed Language of φ is defined as $L_{pt}(\varphi) = \{\rho, i \mid \rho, i \models \varphi\}$. A TPTL formula is said to be closed if every variable is quantified using freeze quantifier before it appears in a clock constraint. For example, $x.y.(a\mathsf{U}(b \wedge x \in (1,2) \wedge y \in (2,3)))$ is a closed formula while $x.(a \wedge y \in (2,3))\mathsf{U}y.(b \wedge x \in (1,2))$ is not closed (or open), as y is used in a clock constraint before it is frozen. Note that for a closed formula, the satisfaction of the model is independent of the clock valuation. In other words, if ψ is a closed formula, then either for every valuation ν, $\rho, i, \nu \models \psi$; or for every valuation ν, $\rho, i, \nu \not\models \psi$. Hence, for a closed formula ψ, we drop the valuation tuple while evaluating for satisfaction as $\rho, i, \nu \models \psi$ for any valuation ν, iff $\rho, i, \overline{0} \models \psi$.

Logic 1-TPTL: The subclass of TPTL that uses **only 1 clock variable** (i.e., $|X| = 1$) is known as 1-TPTL. As an example, the closed formula $\varphi = x.(a\mathsf{U}(b\mathsf{U}(c \wedge T - x \in [1,2])))$ is satisfied by the timed word $\rho = (a, 0)(a, 0.2)(b, 1.1)(b, 1.9)(c, 1.91)(c, 2.1)$, since $\rho, 1 \models \varphi$. The word $\rho' = (a, 0)(a, 0.3)(b, 1.4)(c, 2.1)(c, 2.5)$ does not satisfy φ. However, $\rho', 2 \models \varphi$: If we start from the second position of ρ', then we assign $\nu(x) = 0.3$, and when we reach the position 4 of ρ' with $\tau_4 = 2.1$, we obtain $T - x = 2.1 - 0.3 \in [1,2]$. Note that an MTL$[\mathsf{U}, \mathsf{S}]$ formula can straightforwardly be translated to an equivalent 1-TPTL$[\mathsf{U}, \mathsf{S}]$ (closed) formula. Hence, by Theorem 2.1, we get that the satisfiability checking for 1-TPTL$[\mathsf{U}, \mathsf{S}]$ is undecidable.

Notation: Let x denote the unique freeze variable we use in 1-TPTL. All constraints in 1-*TPTL* have the form $T - x \in I$. (Note that $x - T \in I$ is equivalent to $T - x \in -I$.) Thus, for 1-*TPTL*, let $\widehat{I}$ abbreviate $T - x \in I$.

2.4 Expressive Completeness and Strong Equivalence

Given any specification (formula or automaton) X and Y, X is *equivalent* to Y when for any pointed timed word ρ, i, $\rho, i \models X \iff \rho, i \models Y$. We say that a formalism $\mathcal{X}$ (logic or machine) is *expressively complete* to $\mathcal{Y}$, denoted by $\mathcal{Y} \subseteq \mathcal{X}$, if and only if, for any formulae/automata $Y \in \mathcal{Y}$ there exists an equivalent $X \in \mathcal{X}$. $\mathcal{X}$ is said to be *expressively equivalent* to $\mathcal{Y}$, denoted by $\mathcal{X} \cong \mathcal{Y}$, when $\mathcal{X} \subseteq \mathcal{Y}$ and $\mathcal{Y} \subseteq \mathcal{X}$.

3 INTRODUCING NON-ADJACENT 1-TPTL AND PNUELI EMTL

In this section, we define non-adjacent 1-TPTL. We also give a generalization of MTL called PnEMTL and define its non-adjacent fragment. Let x denote the unique freeze variable we use in 1-TPTL.

3.1 Non-adjacent 1-TPTL

Non-Adjacent 1-TPTL (NA-1-TPTL) is defined as a subclass of 1-TPTL where adjacent intervals within the scope of any freeze quantifier is disallowed. Two intervals $I_1, I_2 \in \mathcal{I}_{\mathrm{int}}$ are non-adjacent iff $\sup(I_1) \neq \inf(I_2) \lor \sup(I_1) = 0$. A set $\mathcal{I}_\nu$ of intervals is non-adjacent iff any two intervals in $\mathcal{I}_\nu$ are non-adjacent. It does not contain punctual intervals other than $[0, 0]$, as every punctual interval is adjacent to itself. For example, the set $\{[1, 2), (2, 3], [5, 6)\}$ is not a non-adjacent set, while $\{[0, 0], [0, 1), (3, 4], [5, 6)\}$ is. Let $\mathcal{I}_{na}$ denote a set of non-adjacent intervals with end points in $\mathbb{Z} \cup \{-\infty, \infty\}$. Consider the following example of a formula in non-adjacent 1-TPTL:

Example 3.1 (Non-adjacent 1-TPTL). An indoor cycling exercise regime may be specified as follows: One must slow-pedal (prop. *sp*) for at least 60 seconds but until the odometer reads 1 km (prop. *od1*). From then onwards one must fast-pedal (prop *fp*) to a time point in the interval [600, 900] from the start of the exercise such that pulse rate is sufficiently high (prop *ph*) for the last 60 seconds of the exercise. This can be given by the following formula:

$$x.sp \, \mathsf{U} \left[\begin{array}{l} \widehat{[60, \infty)} \land od1 \land \\ (\, fp \, \mathsf{U} \, (\widehat{[600, 900]} \land x.H(\widehat{[-60, 0]} \Rightarrow ph)) \,) \end{array} \right].$$

It can be shown that this formula cannot be expressed in logic MITL.

The freeze depth of a TPTL formula φ, $\mathsf{fd}(\varphi)$ is defined inductively. For a propositional formula $prop$, $\mathsf{fd}(prop) = 0$. Also, $\mathsf{fd}(x.\varphi) = \mathsf{fd}(\varphi) + 1$, and $\mathsf{fd}(\varphi_1 \mathsf{U} \varphi_2) = \mathsf{fd}(\varphi_1 \mathsf{S} \varphi_2) = \mathsf{fd}(\varphi_1 \land \varphi_2) = \mathsf{fd}(\varphi_1 \lor \varphi_2) = \mathsf{Max}(\mathsf{fd}(\varphi_1), \mathsf{fd}(\varphi_2)), \mathsf{fd}(\mathcal{G}(\varphi)) = \mathsf{fd}(\mathcal{H}(\varphi)) = \mathsf{fd}(\varphi)$.

Theorem 3.2. *Non-adjacent 1-TPTL*$[\mathsf{U}, \mathsf{S}]$ *is more expressive than MITL*$[\mathsf{U}, \mathsf{S}]$*. It can also express the Counting and the Pnueli modalities of References [25, 26].*

Proof. The straightforward translation of MITL into TPTL in fact gives rise to non-adjacent 1-$TPTL$ formula, e.g., MITL formula $a\mathsf{U}_{[2,3]}(b\mathsf{U}_{[3,4]}c)$ translates to $x.(a\mathsf{U}(\widehat{[2,3]} \land x.(b\mathsf{U}(\widehat{[3,4]} \land c)))$. It has been previously shown that $\mathsf{F}[x.(a \land \mathsf{F}(b \land \widehat{(1,2)} \land \mathsf{F}(c \land \widehat{(1,2)})))]$, which is in fact a formula of non-adjacent 1-$TPTL$, is inexpressible in MTL$[\mathsf{U}, \mathsf{S}]$ (see Reference [39]). The Pnueli modality $\mathsf{Pn}_I(\phi_1, \ldots, \phi_k)$ expresses that there exist positions $i_1 \leq \cdots \leq i_k$ within (relative) interval I where each i_j satisfies ϕ_j. This is equivalent to the non-adjacent 1-$TPTL$ formula $x.(\mathsf{F}(\hat{I} \land \phi_1 \land \mathsf{F}(\hat{I} \land \phi_2 \land \mathsf{F}(\ldots))))$. Similarly, the (simpler) counting modality can also be expressed. □

3.2 Pnueli Automata Modalities

There have been several attempts to extend logic MTL with regular expression/automaton modalities [17, 27, 32, 46]. One of the most general amongst these is Automata Modalities, proposed by

Wilke [46]. MITL (or MTL) extended with these automata modalities was called EMITL (or EMTL, respectively). We further generalize these automata modalities to give automata modalities of arbitrary arity. We call these modalities as *Pnueli Automata Modalities*. The extension is in the same spirit as the extension of future and past modalities to Pnueli future and Pnueli Past modalities in Reference [26]. We call **MTL extended with these Pnueli Automata Modalities** as **PnEMTL**. We now first introduce EMTL before introducing PnEMTL for the sake of readability. For any finite automaton A, let $L(A)$ denote the language of A.

3.2.1 MTL Extended with Automata Modalities, EMTL. Given a finite alphabet Σ, formulae of EMTL have the following syntax:
$\varphi ::= a \mid \varphi \wedge \varphi \mid \neg\varphi \mid \mathcal{F}_I(\mathsf{A})(S) \mid \mathcal{P}_I(\mathsf{A})(S)$ where $a \in \Sigma$, $I \in \mathcal{I}_{\text{nat}}$ and A is an automata over 2^S where S is a set of formulae from EMTL. $\mathcal{F}_I$ and $\mathcal{P}_I$ are future and past **Automata** Modalities, respectively.

Let $\rho = (\sigma_1, \tau_1), \ldots (\sigma_n, \tau_n) \in T\Sigma^*$, $x, y \in dom(\rho)$, $x \leq y$ and $S = \{\varphi_1, \ldots, \varphi_n\}$ be a given set of EMTL subformulae. Let S_i be the exact subset of formulae from S evaluating to true at ρ, i, and let $\text{Seg}^+(\rho, x, y, S)$ and $\text{Seg}^-(\rho, y, x, S)$ be the untimed words $S_x S_{x+1} \ldots S_y$ and $S_y S_{y-1} \ldots S_x$, respectively. Then, the satisfaction relation for ρ, i_0 satisfying a EMTL formula φ is defined recursively as follows:

- $\rho, i_0 \models \mathcal{F}_I(\mathsf{A})(S)$ iff $\exists i_0 \leq i_1 \leq n$ s.t. $[(\tau_{i_1} - \tau_{i_0} \in I_1) \wedge \text{Seg}^+(\rho, i_0, i_1, S) \in L(\mathsf{A})]$,
- $\rho, i_0 \models \mathcal{P}_I(\mathsf{A})(S)$ iff $\exists i_0 \geq i_1 \geq 1$ s.t. $[(\tau_{i_0} - \tau_{i_1} \in I_1) \wedge \text{Seg}^-(\rho, i_0, i_1, S) \in L(\mathsf{A})]$.

Language of any EMTL formula φ, $L(\varphi) = \{\rho \mid \rho, 1 \models \varphi\}$. The Pointed Language of φ is defined as $L_{pt}(\varphi) = \{\rho, i \mid \rho, i \models \varphi\}$. Logic EMITL is a sublogic of EMTL where only non-punctual intervals are allowed along with the modalities $\mathcal{F}$ a and $\mathcal{P}$. Similarly, $\text{EMITL}_{0,\infty}$ is defined as a sublogic of EMITL where the timing intervals associated with both the modalities is restricted to be either of the form $\langle 0, u\rangle$ or of the form $\langle l, \infty)$ where l and u are any non-negative integers.

THEOREM 3.3. *Satisfiability Checking for EMITL is decidable [46] with EXPSPACE complete [17, 27]. Moreover, satisfiability checking for $\text{EMITL}_{0,\infty}$ is PSPACE complete [27].*

3.2.2 MTL Extended with Pnueli Automata Modalities, PnEMTL. PnEMTL is defined similarly as EMTL. Given a finite alphabet Σ, formulae of PnEMTL have the following syntax:
$\varphi ::= a \mid \varphi \wedge \varphi \mid \neg\varphi \mid \mathcal{F}^k_{I_1,\ldots,I_k}(\mathsf{A}_1, \ldots, \mathsf{A}_{k+1})(S) \mid \mathcal{P}^k_{I_1,\ldots,I_k}(\mathsf{A}_1, \ldots, \mathsf{A}_{k+1})(S)$ where $a \in \Sigma$, $I_1, I_2, \ldots I_k \in \mathcal{I}_{\text{nat}}$ and $\mathsf{A}_1, \ldots \mathsf{A}_{k+1}$ are automata over 2^S where S is a set of formulae from PnEMTL. $\mathcal{F}^k$ and $\mathcal{P}^k$ are the new modalities called future and past **Pnueli Automata** Modalities, respectively, where k is the arity of these modalities.

Let $\rho = (\sigma_1, \tau_1), \ldots (\sigma_n, \tau_n) \in T\Sigma^*$, $x, y \in dom(\rho)$, $x \leq y$ and $S = \{\varphi_1, \ldots, \varphi_n\}$ be a given set of PnEMTL formulae. Let $\text{Seg}^+(\rho, x, y, S)$ and $\text{Seg}^-(\rho, y, x, S)$ be as defined previously. Then, the satisfaction relation for ρ, i_0 satisfying a PnEMTL formula φ is defined recursively as follows:

- $\rho, i_0 \models \mathcal{F}^k_{I_1,\ldots,I_k}(\mathsf{A}_1, \ldots, \mathsf{A}_{k+1})(S)$ iff $\exists i_0 \leq i_1 \leq i_2 \ldots \leq i_k \leq n$ s.t.
$\bigwedge_{w=1}^{k} [(\tau_{i_w} - \tau_{i_0} \in I_w) \wedge \text{Seg}^+(\rho, i_{w-1}, i_w, S) \in L(\mathsf{A}_w)] \wedge \text{Seg}^+(\rho, i_k, n, S) \in L(\mathsf{A}_{k+1})$,
- $\rho, i_0 \models \mathcal{P}^k_{I_1,I_2,\ldots,I_k}(\mathsf{A}_1, \ldots, \mathsf{A}_k, \mathsf{A}_{k+1})(S)$ iff $\exists i_0 \geq i_1 \geq i_2 \ldots \geq i_k \geq 1$ s.t.
$\bigwedge_{w=1}^{k} [(\tau_{i_0} - \tau_{i_w} \in I_w) \wedge \text{Seg}^-(\rho, i_{w-1}, i_w, S) \in L(\mathsf{A}_w)] \wedge \text{Seg}^-(\rho, i_k, 1, S) \in L(\mathsf{A}_{k+1})$.

Refer to Figure 1 for semantics of $\mathcal{F}^k$.

Language of any PnEMTL formula φ, as $L(\varphi) = \{\rho \mid \rho, 1 \models \varphi\}$. The Pointed Language of φ is defined as $L_{pt}(\varphi) = \{\rho, i \mid \rho, i \models \varphi\}$. Given a PnEMTL formula φ, its arity is the maximum number of intervals appearing in any $\mathcal{F}, \mathcal{P}$ modality of φ. For example, the arity of $\varphi = \mathcal{F}^2_{I_1,I_2}(\mathsf{A}_1, \mathsf{A}_2, \mathsf{A}_3)(S_1) \wedge \mathcal{P}^1_{I_1}(\mathsf{A}_1, \mathsf{A}_2)(S_2)$ for some sets of formulae S_1, S_2 is 2. For the sake of brevity, $\mathcal{F}^k_{I_1,\ldots,I_k}(\mathsf{A}_1, \ldots, \mathsf{A}_k)(S)$ denotes $\mathcal{F}^k_{I_1,\ldots,I_k}(\mathsf{A}_1, \ldots, \mathsf{A}_k, \mathsf{A}_{k+1})(S)$ where automata A_{k+1}

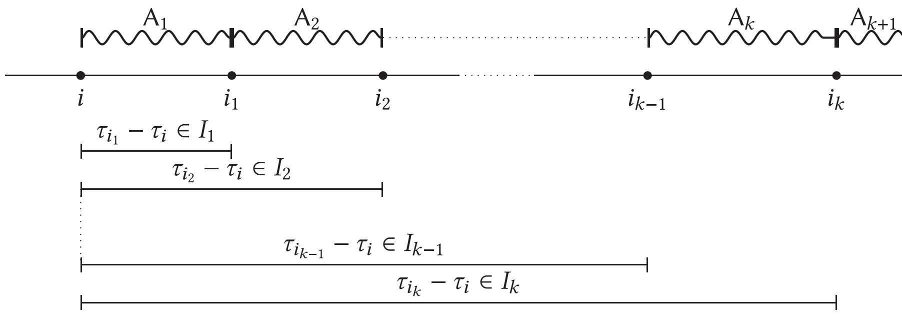

Fig. 1. Figure showing semantics of $\mathcal{F}^k_{I_1,\ldots,I_k}(\mathsf{A}_1, \mathsf{A}_2, \ldots, \mathsf{A}_k, \mathsf{A}_{k+1})(S)$.

accepts all the strings over S. We define **non-adjacent PnEMTL (NA-PnEMTL)** as a subclass where every modality $\mathcal{F}^k_{\mathsf{I}_1,\ldots,\mathsf{I}_k}$ and $\mathcal{P}^k_{\mathsf{I}_1,\ldots,\mathsf{I}_k}$ is such that $\{\mathsf{I}_1, \ldots, \mathsf{I}_k\}$ is a non-adjacent set of intervals.

Note that EMITL of Reference [46] (and variants of it studied in References [27, 32, 33]) are special cases of the non-adjacent PnEMTL modality where the arity is restricted to 1 and the second automata in the argument accepts all the strings. Hence, automaton modality of Reference [46] is of the form $\mathcal{F}_{\mathsf{I}}(A)(S)$ and $\mathcal{P}_I(A)(S)$. Following is an example of a specification that could be naturally written as non-adjacent PnEMTL formula.

Example 3.4 (Non-adjacent PnEMTL). A sugar-level test involves the following: A patient visits the lab and is given a sugar measurement test (prop *sm*) to get fasting sugar level. After this, she is given glucose (prop *gl*) and this must be within 5 min of coming to the lab. After this, the patient rests between 120 and 150 minutes and she is administered sugar measurement again to check the sugar clearance level. Following this, the result (prop *rez*) is given out between 23 to 25 hours (1,380, 1,500 min) of coming to the lab. We assume that these propositions are mutually exclusive and prop *idle* denotes negation of all of them. This protocol is specified by the following non-adjacent PnEMTL formula. For convenience, we specify the automata by their regular expressions. We follow the convention where the tail automaton A_{k+1} can be omitted in $\mathcal{F}^k$.

$$\mathcal{F}^2_{[0,5],\ [1,380,1,500]} \left[\begin{array}{l} sm \cdot (idle^*) \cdot (gl \wedge \mathcal{F}^1_{[120,150]}(\ gl \cdot (idle^*) \cdot sm\ ), \\ gl \cdot ((\neg rez)^*) \cdot rez \end{array} \right]$$

For readability, the two regular expressions of the top $\mathcal{F}^2$ are given in two separate lines. It states that the first regular expression must end at time within $[0, 5]$ of starting and the second regular expression must end at a time within [1,380, 1,500] of starting. Note the nested use of $\mathcal{F}$ to anchor the duration between glucose and the second sugar measurement.

3.3 Size of Formulae

The size of a temporal logic formula can be measured as usual, using the parse tree of the formula, or using the parse **DAG (Directed Acyclic Graph)** of the formula, where a syntactically unique subformula occurs only once. The latter representation is more succinct and is used widely starting from the classical LTL formula-to-automaton construction [14, 45]. For our results also, we will use the notion of DAG-size of a formula.

The (DAG) size of a formula φ denoted by $|\varphi|$ is a measure of how many bits are required to store it in the DAG representation. The size of a TPTL formula is defined as the sum of the number of U, S and Boolean operators and freeze quantifiers in it. For PnEMTL formulae, $|op|$ is defined as the number of Boolean operators and variables used in it. $|(\mathcal{F}^k_{I_1,\ldots,I_k}(\mathsf{A}_1, \ldots, \mathsf{A}_{k+1})(S)| = \sum_{\varphi \in S}(|\varphi|) + |\mathsf{A}_1| + \cdots + |\mathsf{A}_{k+1}| + 2k \times log(\text{cmax})$ where $|\mathsf{A}|$ denotes the size of the automaton A given by the sum of number of its states and transitions and cmax denotes the maximum allowable value of the constant used in the intervals $I_1, \dots, I_k$.[6]

4 ANCHORED INTERVAL WORD ABSTRACTION

All the logics considered here have the feature that a sub-formula asserts timing constraints on various positions relative to an anchor position; e.g., the position of freezing the clock in TPTL. Such constraints can be symbolically represented as an interval word with a unique anchor position and all other positions carry a set of time intervals constraining the timestamp of the position relative to the timestamp of the anchor. See interval word κ in Figure 2. We now define these interval words formally. Let $\mathcal{I}_\nu \subseteq \mathcal{I}_{\mathsf{int}}$. An I_ν-interval word over Σ is a word κ of the form $\sigma_1\sigma_2 \dots \sigma_n \in (2^{\Sigma \cup \{\mathsf{anch}\} \cup \mathcal{I}_\nu})^*$ such that:

(1) There is a unique $i \in dom(\kappa)$ such that $\mathsf{anch} \in \sigma_i$. Such a position is called the *anchor* of κ and denoted by $\mathsf{anch}(\kappa)$.
(2) At all the points in κ, at least one of the propositions from Σ holds. That is, for all $i \in dom(\kappa)$, $\sigma_i \cap \Sigma$ is a non-empty set.

Let J be any interval in $\mathcal{I}_\nu$. We say that a point $i \in dom(\kappa)$ is a J-time-restricted point if and only if, $J \in \sigma_i$. i is called time-restricted point if and only if either i is J-time restricted for some interval J in $\mathcal{I}_\nu$ or $\mathsf{anch} \in a_i$.

From I_ν-interval word to Timed Words: Given a $\mathcal{I}_\nu$-interval word $\kappa = \sigma_1 \dots \sigma_n$ over Σ and a timed word $\rho = (\sigma'_1, \tau_1) \dots (\sigma'_m, \tau_m)$, the pointed timed word ρ, i is consistent with κ iff $dom(\rho) = dom(\kappa)$ (i.e., $m = n$), $i = \mathsf{anch}(\kappa)$, for all $j \in dom(\kappa)$, $\sigma'_j = \sigma_j \cap \Sigma$ and, $I \in \sigma_j \cap \mathcal{I}_\nu$ implies $\tau_j - \tau_i \in I$. Thus, κ and ρ, i agree on propositions from Σ at all positions, and the timestamp of any position j in ρ satisfies every interval constraint in σ_j relative to τ_i, the timestamp of anchor position. $\mathsf{Time}(\kappa)$ denotes the set of all the pointed timed words consistent with a given interval word κ, and $\mathsf{Time}(\Omega) = \bigcup_{\kappa \in \Omega}(\mathsf{Time}(\kappa))$ for a set of interval words Ω. Note that the "consistency relation" is a many-to-many relation.

Example 4.1. Let $\kappa = \{a, b, (-1, 0)\}\{b, (-1, 0)\}\{a, \mathsf{anch}\}\{b, [2, 3]\}$ be an interval word over the set of intervals $\{(-1, 0), [2, 3]\}$. Consider timed words ρ and ρ' s.t. $\rho = (\{a, b\}, 0)(\{b\}, .5), (\{a\}, .95)(\{b\}, 3)$, $\rho' = (\{a, b\}, 0)(\{b\}, 0.8)(\{a\}, 0.9)(\{b\}, 2.9)$. Then, $\rho, 3$ as well as $\rho', 3$ are consistent with κ while $\rho, 2$ is not. Likewise, for the timed word $\rho'' = (\{a, b\}, 0), (\{b\}, 0.5), (\{a\}, 1.1)(\{b\}, 3)$, $\rho'', 3$ is not consistent with κ as $\tau_1 - \tau_3 \notin (-1, 0)$, as also $\tau_4 - \tau_3 \notin [2, 3]$.

Let $\mathcal{I}_\nu, \mathcal{I}'_\nu \subseteq \mathcal{I}_{\mathsf{int}}$. Let $\kappa = \sigma_1 \dots \sigma_n$ and $\kappa' = \sigma'_1 \dots \sigma'_m$ be $\mathcal{I}_\nu$ and $\mathcal{I}'_\nu$-interval words, respectively. κ is *similar* to κ', denoted by $\kappa \sim \kappa'$ if and only if, (i) $dom(\kappa) = dom(\kappa')$, (ii) for all $i \in dom(\kappa)$, $a_i \cap \Sigma = b_i \cap \Sigma$, and (iii) $\mathsf{anch}(\kappa) = \mathsf{anch}(\kappa')$. Additionally, κ is *congruent* to κ', denoted by $\kappa \cong \kappa'$, iff $\mathsf{Time}(\kappa) = \mathsf{Time}(\kappa')$. That is, κ and κ' abstract the same set of pointed timed words.

Collapsed Interval Words: The set of interval constraints at a position can be collapsed into a single interval by taking the intersection of all the intervals at that position giving a Collapsed Interval Word. Given an $\mathcal{I}_\nu$-interval word $\kappa = \sigma_1 \dots \sigma_n$, let $I_j = \sigma_j \cap I_\nu$. Let $\kappa' = \mathsf{Col}(\kappa)$ be the word obtained by replacing I_j with $\bigcap_{I \in I_j} I$ in σ_j, for all $j \in dom(\kappa)$. Note that κ' is an interval word over $\mathsf{CL}(\mathcal{I}_\nu) = \{I | I = \bigcap I', I' \subseteq \mathcal{I}_\nu\}$. Note that, if for any j, the set I_j contains two disjoint intervals (like $[1, 2]$ and $[3, 4]$), then $\mathsf{Col}(\kappa)$ is undefined. It is clear that $\mathsf{Time}(\kappa) = \mathsf{Time}(\kappa')$. An interval word κ is called *collapsed* iff $\kappa = \mathsf{Col}(\kappa)$.

[6]Note that we assume that the constants are encoded in binary.

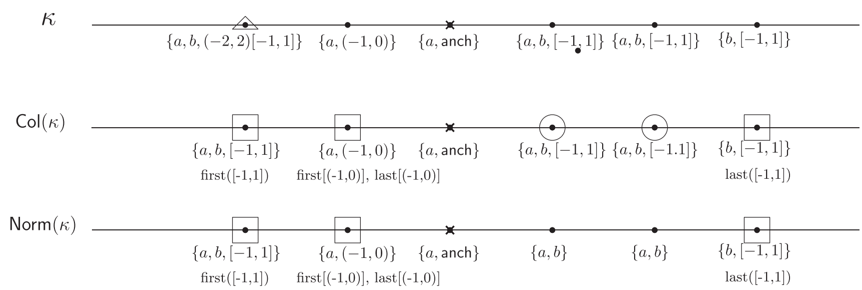

Fig. 2. Point within the triangle has more than one interval. The encircled points are intermediate points and carry redundant information. The required timing constraint is encoded by first and last time-restricted points of all the intervals (within boxes).

Normalization of Interval Words: An interval I may repeat many times in a collapsed interval word κ. Some of these occurrences are redundant, and we can keep only the first and last occurrence of the interval without changing the set of pointed timed words consistent with it hence giving a normalized form of κ. See Figure 2. For a collapsed interval word κ and any $I \in I_\nu$, let $\mathsf{first}(\kappa, I)$ and $\mathsf{last}(\kappa, I)$ denote the positions of first and last occurrence of I in κ. If κ does not contain any occurrence of I, then both $\mathsf{first}(\kappa, I) = \mathsf{last}(\kappa, I) = \bot$. We define, $\mathsf{Boundary}(\kappa) = \{i | i \in dom(\kappa) \land \exists I \in I_\nu \text{ s.t. } (i = \mathsf{first}(\kappa, I) \lor i = \mathsf{last}(\kappa, I) \lor i = \mathsf{anch}(\kappa))\}$.

The normalized interval word corresponding to κ, denoted $\mathsf{Norm}(\kappa)$, is defined as $\kappa_{nor} = \sigma'_1 \ldots \sigma'_m$, such that (i) $\kappa_{nor} \sim \mathsf{Col}(\kappa)$, (ii) for all $I \in \mathsf{CL}(I_\nu)$, $\mathsf{first}(\kappa, I) = \mathsf{first}(\kappa_{nor}, I)$, $\mathsf{last}(\kappa, I) = \mathsf{last}(\kappa_{nor}, I)$, and for all points $j \in dom(\kappa_{nor})$ with $\mathsf{first}(\kappa, I) < j < \mathsf{last}(\kappa, I)$, j is not a I-time-constrained point. See Figure 2. Hence, a normalized word is a collapsed word where for any $J \in \mathsf{CL}(I_\nu)$ there are at most two J-time-restricted points. This is the key property that will be used to reduce 1-TPTL to a (bounded arity) PnEMTL formulae. In what follows, for any interval work $\kappa = \sigma_1 \ldots \sigma_n$, for any point $j \in dom(\kappa)$, $\kappa[j] = \sigma_j$. Similarly, for any timed word $\rho = (\sigma'_1, \tau_1) \ldots (\sigma'_m, \tau_m)$, for any $j \in dom(\rho)$, $\rho[j] = (\sigma_j, \tau_j)$, $\rho[j](1) = \sigma_j$ and $\rho[j](2) = \tau_j$.

The proof follows from the fact that $\kappa \cong \mathsf{Col}(\kappa)$ and, since $\mathsf{Col}(\kappa) \sim \mathsf{Norm}(\kappa)$, the set of timed words consistent with any of them will have identical untimed behavior. For the timed part, the key observation is as follows: For some interval $I \in I_\nu$, let $i' = \mathsf{first}(\kappa, I), j' = \mathsf{last}(\kappa, I)$. Then, for any ρ, i in $\mathsf{Time}(\kappa)$, points i' and j' are within the interval I from point i. Hence, any point $i' \le i'' \le j'$ is also within interval I from i. Thus, the interval I need not be explicitly mentioned at intermediate points. Formally, the following two lemmas Lemma 4.2 and Lemma 4.3 imply Lemma 4.4. Lemma 4.2 shows $\kappa \cong \mathsf{Col}(\kappa)$. Lemma 4.3 implies that $\mathsf{Col}(\kappa) \cong \mathsf{Norm}(\kappa)$.

Lemma 4.2. *Let κ be an I_ν-interval word. Then, $\kappa \cong \mathsf{Col}(\kappa)$.*

Proof. A pointed word ρ, i is consistent with κ iff

(i) $dom(\rho) = dom(\kappa)$,
(ii) $i = \mathsf{anch}(\kappa)$,
(iii) for all $j \in dom(\kappa)$, $\rho[j](1) = \kappa[j] \cap \Sigma$ and
(iv) for all $j \neq i$, $I \in a_j \cap I_\nu$ implies $\rho[j](2) - \rho[i](2) \in I$.
(v) $\kappa \sim \mathsf{Col}(\kappa)$, by definition of Col.

Hence, given (v), (i) iff (a) (ii) iff (b) (iii) iff (c) where:
(a) $dom(\rho) = dom(\kappa) = dom(\mathsf{Col}(\kappa))$, (b) $i = \mathsf{anch}(\kappa) = \mathsf{anch}(\mathsf{Col}(\kappa))$, (c) for all $j \in dom(\kappa)$, $\rho[j](1) = \kappa[j] \cap \Sigma = \mathsf{Col}(\kappa)[j] \cap \Sigma$. (iv) is equivalent to $\rho[j](2) - \rho[i](2) \in \bigcap(\kappa[j] \cap I_\nu)$, but $\bigcap(\kappa[j] \cap I_\nu) = \mathsf{Col}(\kappa)[j]$. Hence, (iv) iff (d) $\rho[j](2) - \rho[i](2) \in \mathsf{Col}(\kappa)[j]$. Hence, (i), (ii), (iii), and (iv) iff (a), (b), (c), and (d). Hence, ρ, i is consistent with κ iff it is consistent with $\mathsf{Col}(\kappa)$. □

Lemma 4.3. *Let κ and κ' be I_ν-interval words such that $\kappa \sim \kappa'$. If for all $I \in I_\nu$,* $\mathsf{first}(\kappa, I) = \mathsf{first}(\kappa', I)$ *and* $\mathsf{last}(\kappa, I) = \mathsf{last}(\kappa', I)$*, then $\kappa \cong \kappa'$.*

Proof. The proof idea is the following:

- As $\kappa \sim \kappa'$, the set of timed words consistent with any of them will have identical untimed behavior.
- For the timed part, the key observation is as follows: For some interval $I \in I_\nu$, let $i' = \mathsf{first}(\kappa, I), j' = \mathsf{last}(\kappa, I)$. Then, for any ρ, i in $\mathsf{Time}(\kappa)$, points i' and j' are within the interval I from point i. Hence, any point $i' \leq i'' \leq j'$ is also within interval I from i. Thus, the intermediate I-time-restricted points (I-time-restricted points other than the first and the last) do not offer any extra information regarding the timing behavior. In other words, the restriction from the first and last I restricted points will imply the restrictions offered by intermediate I restricted points. Hence, their presence or absence makes no difference.

Both Lemmas 4.2 and 4.3 imply the following lemma, which will be used in the reduction of 1-TPTL to PnEMTL:

Lemma 4.4. $\kappa \cong \mathsf{Norm}(\kappa)$. *Note,* $\mathsf{Norm}(\kappa)$ *has at most* $2 \times |I_\nu|^2 + 1$ *restricted points.*

Let $\rho = (a_1, \tau_1), \dots (a_n, \tau_n)$ be any timed word. ρ, i is consistent with κ iff

(1) (i) $dom(\rho) = dom(\kappa)$,
(ii) $i = \mathsf{anch}(\kappa)$,
(iii) for all $j \in dom(\rho)$, $\kappa[j] \cap \Sigma = a_j$ and
(iv) for all $j \neq i \in dom(\rho)$, $\tau_j - \tau_i \in \bigcap(I_\nu \cap \kappa[j])$.
Similarly, ρ, i is consistent with κ' if and only if
(2)(a) $dom(\rho) = dom(\kappa')$,
(b) $i = \mathsf{anch}(\kappa')$,
(c) for all $j \in dom(\rho)$, if $\kappa'[j] \cap \Sigma = a_j$ and
(d) for all $j \neq i \in dom(\rho)$, $\tau_j - \tau_i \in \bigcap(I_\nu \cap \kappa'[j])$.

Note that, as $\kappa \sim \kappa'$, we have, $dom(\kappa) = dom(\kappa')$, $\mathsf{anch}(\kappa) = \mathsf{anch}(\kappa')$, for all $j \in dom(\kappa)$, $\kappa[j] \cap \Sigma = \kappa'[j] \cap \Sigma$. Thus, 2(a) $\equiv$ 1(i), 2(b) $\equiv$ 1(ii), and 2(c) $\equiv$ 1(iii).

Suppose there exists a ρ, i consistent with κ but there exists $j' \neq i \in dom(\rho)$, $\tau_j' - \tau_i \notin I'$ for some $I' \in \kappa'[j']$. By definition, $\mathsf{first}(\kappa', I') \leq j' \leq \mathsf{last}(\kappa', I')$. But $\mathsf{first}(\kappa', I') = \mathsf{first}(\kappa, I')$, $\mathsf{last}(\kappa', I') = \mathsf{last}(\kappa, I')$. Hence, $\mathsf{first}(\kappa, I') \leq j' \leq \mathsf{last}(\kappa, I')$. As the timestamps of the timed word increases monotonically, $x \leq y \leq z$ implies that $\tau_x \leq \tau_y \leq \tau_z$, which implies that $\tau_x - \tau_i \leq \tau_y - \tau_i \leq \tau_z - \tau_i$. Hence, $\tau_{\mathsf{first}(\kappa,I')} - \tau_i \leq \tau_{j'} - \tau_i \leq \tau_{\mathsf{last}(\kappa,I')} - \tau_i$. But $\tau_{\mathsf{first}(\kappa,I')} - \tau_i \in I'$ and $\tau_{\mathsf{last}(\kappa,I')} - \tau_i \in I'$, because ρ is consistent with κ. This implies that $\tau_{j'} - \tau_i \in I'$ (as I' is a convex set), which is a contradiction. Hence, if ρ, i is consistent with κ, then it is consistent with κ', too. By symmetry, if ρ, i is consistent with κ', then it is also consistent with κ. Hence, $\kappa \cong \kappa'$. □

We give a road map to the proofs of results in Figure 3. In summary,

$$\text{1-TPTL} < \text{GQMSO} \equiv \text{PnEMTL}, \quad (1)$$
$$\text{NA-1-TPTL} < \text{NA-GQMSO} \equiv \text{NA-PnEMTL}. \quad (2)$$

The logics in (1) all have undecidable satisfiability, whereas logics in (2) all have decidable satisfiability. Specifically, NA-1-TPTL and NA-PnEMTLhave EXPSPACE-complete satisfiability checking.

5 1-TPTL TO PNEMTL

In this section, we reduce a 1-TPTL formula into an equivalent PnEMTL formula. First, we consider 1-TPTL formula in negation normal form with a single outermost freeze quantifier (call these

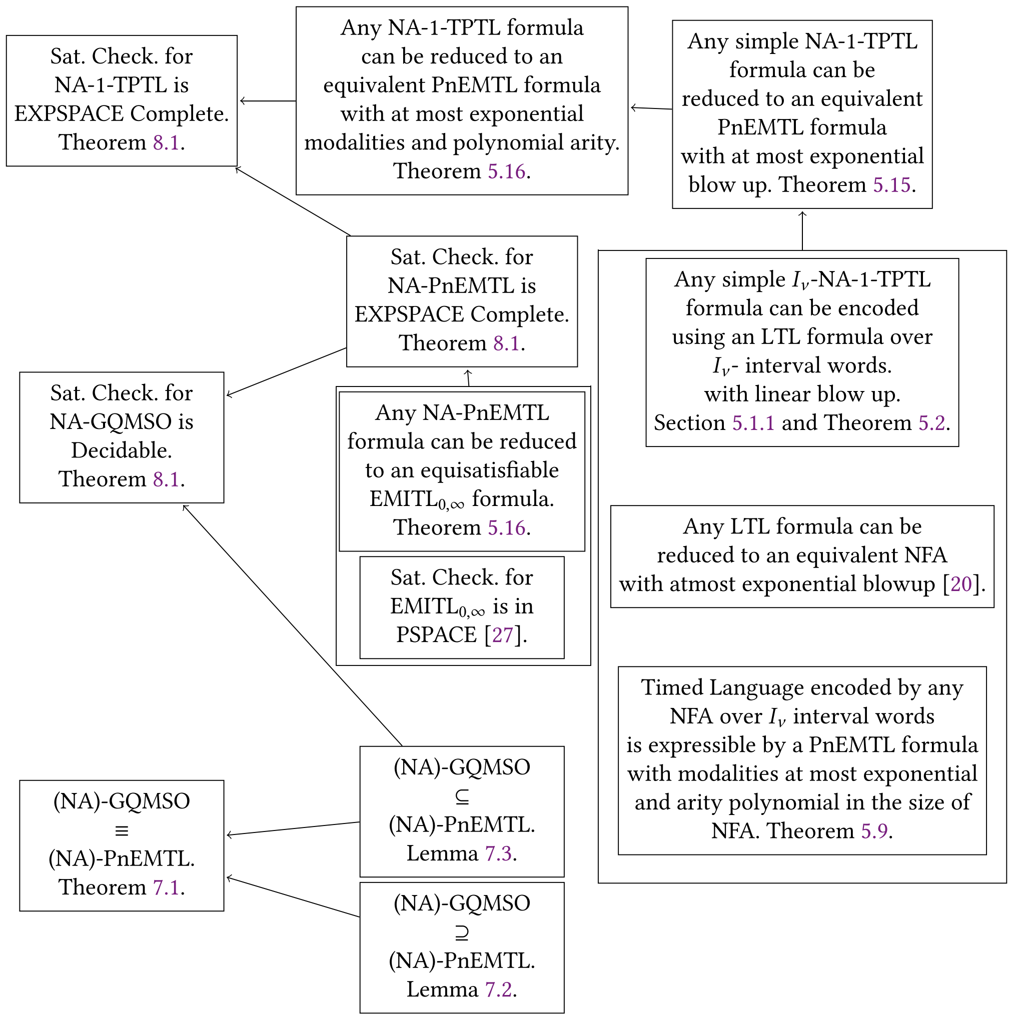

Fig. 3. The above figure is a road map of all the main and intermediate results. An arrow from result A to result B indicates that the proof of result B uses the result A. For definitions of GQMSO and NA-GQMSO, please refer Section 6.

simple TPTL formulae) and give the reduction. More complex formulae can be handled by applying the same reduction (shown below) recursively. For any set of formulae S, let $\bigvee S$ denote $\bigvee_{s \in S} s$. This notation will be extensively used from this point onwards in all the succeeding sections, too. A TPTL formula is said to be *simple* if it is in the negation normal form and of the form $x.\varphi$ where, φ is a 1-TPTL formula with no freeze quantifiers. Let $\psi = x.\varphi$ be a simple TPTL formula. Let $\mathcal{I}_\nu = \{I \mid T - x \in I \text{ or appears in } \varphi\} \cup \{-I \mid x - T \in I \text{ or appears in } \varphi\}$ and let $\mathsf{CL}(\mathcal{I}_\nu) = I_\nu$. We construct a PnEMTL formula ϕ, such that $\rho, i \models \psi \iff \rho, i \models \phi$. We break this construction into the following steps:

(1) We construct an LTL formula α s.t. $L(\alpha)$ contains only $\mathcal{I}_\nu$-interval words and $\rho, i \models \psi$ iff $\rho, i \in \mathsf{Time}(L(\alpha))$. Let A_α be the NFA s.t. $L(A_\alpha) = L(\alpha)$ (constructed using Reference [20]). We then construct NFA, A, over $I_\nu = \mathsf{CL}(\mathcal{I}_\nu)$ interval words from A_α such that $L(A) = \mathsf{Col}(L(A_\alpha))$. Note that $|I_\nu| \le |\mathcal{I}_\nu|^2$. Hence, $\rho, i \models \psi$ iff $\rho, i \in \mathsf{Time}(L(A))$.

(2) Let W be the set of all I_ν-interval words. We can partition W into finitely many types, each *type* capturing a certain relative ordering between first and last occurrences of intervals from I_ν as well as anch. Let $\mathcal{T}(I_\nu)$ be the finite set of all such types. For each type seq $\in \mathcal{T}(I_\nu)$, we construct an NFA, A_{seq}, such that $L(A_{\mathsf{seq}}) = \mathsf{Norm}(L(A) \cap W_{\mathsf{seq}})$, where W_{seq} is the set of all the I_ν-interval words of type seq. Hence, A_{seq} accepts only normalized interval words of type seq.

(3) For every type seq, using the A_{seq} above, we construct a PnEMTL formula ϕ_{seq} such that, $\rho, i \models \phi_{\mathsf{seq}}$ if and only if $\rho, i \in \mathsf{Time}(L(A_{\mathsf{seq}}))$. The desired $\phi = \bigvee_{\mathsf{seq} \in \mathcal{T}(I_\nu)} \phi_{\mathsf{seq}}$. Hence, $L_{pt}(\phi) = \bigcup_{\mathsf{seq} \in \mathcal{T}(I_\nu)} \mathsf{Time}(L(A_{\mathsf{seq}})) = \mathsf{Time}(L(A)) = L_{pt}(\psi)$.

We suggest the reader to refer to our running example (Examples 5.1, 5.7, 5.8, 5.11, 5.14) for step-by-step reduction of simple 1-TPTL formula to PnEMTL formula, Section 5.1. Example 5.1 gives reduction from simple 1-TPTL formula, ψ, to an LTL formula, α, over interval words. Example 5.7 (Figure 4) gives the automaton, A_α, over interval words equivalent to α constructed in Example 5.1. Example 5.8 (Figure 5) gives the construction of automaton, A, over collapsed interval words from A_α constructed in Example 5.7. Example 5.11 (Figure 6) gives the construction of a normalized automaton, A_{seq}, for type one particular type seq from automaton A. Finally, Example 5.14 gives a construction of PnEMTL formula ϕ_{seq} equivalent to timed behaviors encoded by automata, A_{seq}. Disjunctions over all possible types seq is the required PnEMTL formula ϕ equivalent to given 1-TPTL formula ψ.

We give a running example (Examples 5.1, 5.7, 5.8, 5.11, 5.14) along with the construction to facilitate readers in understanding the steps of the construction.

5.1 Simple TPTL to NFA over Interval Words

In this section, we elaborate the first step of the reduction.

5.1.1 Simple TPTL to LTL over Interval Words. Let γ be any 1-TPTL formula without any freeze quantifier. We define LTL(γ) as an LTL formula obtained from γ by replacing clock constraints $T - x \in I$ with I and $x - T \in I$ with $-I$.[7] As above, $\psi = x.\varphi$. Consider an LTL formula $\alpha = \mathsf{F}[\mathrm{LTL}(\varphi) \wedge \mathsf{anch} \wedge \neg(\mathsf{F}(\mathsf{anch}) \vee \mathsf{P}(\mathsf{anch}))] \wedge \mathcal{G}(\bigvee \Sigma)$ over $\Sigma' = \Sigma \cup \mathcal{I}_\nu \cup \{\mathsf{anch}\}$, (LTL($\varphi$) is well defined, as φ has no freeze quantifier). Note that all the words in $L(\mathrm{LTL}(\varphi))$ are $\mathcal{I}_\nu$-interval words, as subformula $\mathsf{anch} \wedge \neg(\mathsf{F}(\mathsf{anch}) \vee \mathsf{P}(\mathsf{anch}))$ makes sure anch is true at exactly one point, i.e., the point where LTL(φ) is asserted (condition (1) in definition of $\mathcal{I}_\nu$ interval word) and the conjunct $\mathcal{G}(\bigvee \Sigma)$ makes sure that there is no such point where only propositions from $\mathcal{I}_\nu \cup \{\mathsf{anch}\}$ hold (condition (2) in definition of $\mathcal{I}_\nu$ interval word).

Example 5.1. Let $\psi = x.\varphi$ where $\varphi = [\varphi_a \wedge \mathsf{F}\{b \wedge x \in (1,2) \wedge \mathsf{F}(c \wedge x \in (0,3))\} \wedge \{(a \wedge x \in (-3,0)\mathsf{S}(c \wedge x \in (-3,0))\} \wedge \mathcal{G}(\varphi_a \vee \varphi_b \vee \varphi_c) \wedge \mathcal{H}(\varphi_a \vee \varphi_b \vee \varphi_c)]$, where $\varphi_a = a \wedge \neg b \wedge \neg c, \varphi_b = \neg a \wedge b \wedge \neg c, \varphi_c = \neg a \wedge \neg b \wedge c$. Then, $\mathrm{LTL}(\varphi) = [\mathsf{F}\{\varphi_a \wedge \mathsf{F}(\varphi_b \wedge (1,2) \wedge \mathsf{F}(\varphi_c \wedge (0,3)))\} \wedge \{\varphi_a \wedge (-3,0)\mathsf{S}(\varphi_c \wedge (-3,0))\} \wedge \mathsf{anch} \wedge \neg(\mathsf{F}(\mathsf{anch}) \vee \mathsf{P}(\mathsf{anch}))]$ and $\alpha = [\mathsf{F}\{\mathrm{LTL}(\varphi) \wedge \mathcal{G}(\bigvee \Sigma)\}]$.

Theorem 5.2. *For any timed word ρ, $i \in dom(\rho)$, $\rho, i \models \psi \iff \rho, i \in \mathsf{Time}(L(LTL(\alpha)))$.*

Proof. Note that for any timed word $\rho = (\sigma_1, \tau_1) \ldots (\sigma_n, \tau_n)$ and $i \in dom(\rho)$, $\rho, i, [x =: \tau_i] \models \varphi$ is equivalent to $\rho, i \models \psi$. Moreover, it is straightforward that α accepts all (and only) those words that are valid $\mathcal{I}_\nu$ interval words where the anchor point satisfies LTL(φ). Let κ be any $\mathcal{I}_\nu$-interval word over Σ with $\mathsf{anch}(\kappa) = i$. It suffices to prove the following:

[7] Note, if $I = [a, b)$, then $-I = (-b, -a]$.

(i) If $\kappa, i \models \text{LTL}(\varphi)$, then for all $\rho \in \mathsf{Time}(\kappa)$ $\rho, i \models \psi$.
(ii) If for any timed word ρ, $\rho, i \models \psi$, then there exists some $\mathcal{I}_\nu$-interval word over Σ such that $\rho, i \in \mathsf{Time}(\kappa)$ and $\kappa, i \models \text{LTL}(\varphi)$.

Intuitively, this is because $\text{LTL}(\varphi)$ is asserting similar timing constraints via interval words that is asserted by φ on the timed words directly. Note, (i) and (ii) is implied by Lemma 5.6. Substitute $j = i$ and $\gamma = \varphi$ in Lemma 5.6. Hence, the above theorem can be seen as a corollary of Lemma 5.6 (below). □

We give some interesting properties of interval words in the next two propositions before giving 5.6.

Proposition 5.3. *Let γ be any subformulae of φ. Let κ, κ' be any $\mathcal{I}_\nu$-interval words such that $\kappa' \sim \kappa$ and for any $i \in dom(\kappa)$ $\kappa[i] \subseteq \kappa'[i]$. For any $j \in dom(\kappa)$, if $\kappa, j \models LTL(\gamma)$ then $\kappa', j \models LTL(\gamma)$.*

Proof. Note that γ is in negation normal form. Hence, any subformulae of the form $x \in I$ will never be within the scope of a negation. Hence, γ can never have a subformulae of the form $\neg(x \in I)$. This implies that $\text{LTL}(\gamma)$ can never have a subformulae of the form $\neg I$ for any $I \in \mathcal{I}_\nu$. We apply structural induction on depth of γ. For base case, γ is a propositional logic formula and $\text{LTL}(\gamma)$ is also a propositional logic formula over Σ and the statement holds trivially for any pair of similar $\mathcal{I}_\nu$-interval words.

If $\gamma = x \in I$, then $\text{LTL}(\gamma) = I$. If $\kappa, j \models I$, then $I \in \kappa[j]$. This implies that $I \in \kappa'[j]$ (as $\kappa[j] \subseteq \kappa'[j]$). Hence, $\kappa', j \models \text{LTL}(\gamma)$. Let γ be any formula such that the proposition is true for every subformula of γ (induction hypothesis). If γ is of the form $\gamma_1 \vee \gamma_2$, and if $\kappa, j \models \gamma$, then $\kappa, j \models \gamma_1$ and $\kappa, j \models \gamma_2$. By induction hypothesis, $\kappa', j \models \gamma_1$ and $\kappa', j \models \gamma_2$. Hence, $\kappa', j \models \gamma$. Similar argument holds if γ is of the form $\gamma_1 \vee \gamma_2$.

If γ is of the form $\gamma_1 \mathsf{U} \gamma_2$. If $\kappa, j \models \gamma$, then (a)$\exists j' > j$ such that $\kappa, j' \models \gamma_2$ and $\forall j < j'' < j'$ $\kappa, j'' \models \gamma_1$. (a) along with the induction hypothesis implies, (b)$\exists j' > j$ such that $\kappa', j' \models \gamma_2$ and $\forall j < j'' < j'$ $\kappa', j'' \models \gamma_1$. (b) implies $\kappa', j \models \gamma$. For the case where γ is of the form $\gamma_1 \mathsf{S} \gamma_2$, $\mathcal{G}\gamma'$ or $\mathcal{H}\gamma'$ similar argument holds. □

Proposition 5.4. *Let $\kappa, \kappa', \kappa''$ be $\mathcal{I}_\nu$-interval words such that $\kappa \sim \kappa' \sim \kappa''$ and $\kappa[j] = \kappa'[j] \cup \kappa''[j]$ for any $j \in dom(\kappa)$. Then,* $\mathsf{Time}(\kappa) = \mathsf{Time}(\kappa') \cap \mathsf{Time}(\kappa'')$.

Proof. We need to prove that $\rho, i \in \mathsf{Time}(\kappa)$ iff $\rho, i \in \mathsf{Time}(\kappa')$ and $\rho, i \in \mathsf{Time}(\kappa'')$. For any $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$ and $i \in dom(\rho)$, $\rho, i \in \mathsf{Time}(\kappa') \cap \mathsf{Time}(\kappa'') \iff \forall j \in dom(\rho), \sigma_j = \kappa'[j] \cap \Sigma = \kappa''[j] \cap \Sigma$ (as $\kappa' \sim \kappa''$) and $\tau_j - \tau_i \in I$ for all $I \in (\kappa'[j] \cap \mathcal{I}_\nu) \cup (\kappa''[j] \cap \mathcal{I}_\nu) \iff \forall j \in dom(\rho), \sigma_j = \kappa[j] \cap \Sigma$(as $\kappa \sim \kappa'' \sim \kappa'$) and $\tau_j - \tau_i \in I$ for all $I \in (\kappa[j] \cap \mathcal{I}_\nu)$(as $\kappa[j] = \kappa'[j] \cup \kappa''[j]$) $\iff \rho, i \in \mathsf{Time}(\kappa)$. □

Before giving Proposition 5.6, We need to define the notion of canonical $\mathcal{I}_\nu$ interval word abstraction for a given pointed timed word ρ, i. Let $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$ and $i \in dom(\rho)$.

Definition 5.5 (Canonical Abstraction). An $\mathcal{I}_\nu$ interval word κ is a canonical $\mathcal{I}_\nu$ interval word abstraction of ρ, denoted by $\mathsf{Can}(\mathcal{I}_\nu, \rho, i)$, iff $\rho, i \in \mathsf{Time}(\kappa)$ and for any $j \in dom(\rho)$ and $I \in \mathcal{I}_\nu$, $I \in \kappa[j]$ iff $\tau_j - \tau_i \in I$.

Hence, κ is the tightest abstraction of ρ, i with respect to the set of intervals $\mathcal{I}_\nu$. It is trivial to observe that Can is a well defined function. We now present the main lemma, which implies Theorem 5.2.

Lemma 5.6. *Let γ be any subformula of φ.*

(i) *For any $\mathcal{I}_\nu$-interval word κ and $j \in dom(\kappa)$, $\kappa, j \models LTL(\gamma)$ implies for all $\rho, i \in$* $\mathsf{Time}(\kappa)$, *$\rho, j, [x =: \tau_i] \models \gamma$.*

(ii) *For every timed word* $\rho = (a_1, \tau_1) \dots (a_n, \tau_n)$ *and* $j \in dom(\rho)$, $\rho, j, [x =: \tau_i] \models \gamma$ *implies* $\kappa, j \models LTL(\gamma)$ *where* $\kappa = \mathsf{Can}(I_\nu, \rho, i)$.

Proof. We apply structural induction on γ.

Base Case: For $\gamma = a$ or $\gamma = \neg a$ where $a \in \Sigma$, (i) and (ii) trivially holds as for every interval word $\kappa = \sigma'_1 \dots \sigma'_n$ and timed word $\rho = (\sigma_1, \tau_1) \dots \sigma_n, \tau_n)$, $\rho, i \in \mathsf{Time}(\kappa)$ implies ρ and κ agree on the set of propositions from Σ. That is, $\sigma'_j \cap \Sigma = \sigma_j$. Moreover, for any propositional formulae γ, $\mathrm{LTL}(\gamma) = \gamma$ and the satisfaction of γ only depends on the present point. For $\gamma = T - x \in I$, $\mathrm{LTL}(\gamma) = I$.

Proving (i): $\kappa, j \models \mathrm{LTL}(\gamma)$ would imply $I \in \kappa[j]$. Then, for any $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$, $\rho, i \in \mathsf{Time}(\kappa)$ only if $\tau_j - \tau_i \in I$, which implies that $\rho, j, [x =: \tau_i] \models \gamma$.

Proving (ii): Consider any timed word $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$ such that $\rho, j, [x =: \tau_i] \models \gamma$. Then, by semantics, $\tau_j - \tau_i \in I$. By definition of canonical abstraction if $\kappa = \mathsf{Can}(I_\nu, \rho, i)$, then $I \in \kappa[j]$. Hence, $\kappa, j \models \mathrm{LTL}(\gamma)$. Similar argument holds for $\gamma = x - T \in I$. Note that we do not have to deal with the case $\gamma = \neg(T - x \in I)$ (or $\gamma = \neg((x - T) \in I)$), as the given ψ and hence (all its subformula φ and γ) are in negation normal form. This is an important observation, as the above lemma will fail to hold for $\gamma = \neg(T - x \in I)$. In this case, $\mathrm{LTL}(\gamma) = \neg I$. Hence, all the interval words $\kappa = \sigma'_1 \dots \sigma'_n$ will satisfy $\mathrm{LTL}(\gamma)$ if $I \notin \sigma'_j$. Note that this would not disallow $\mathsf{Time}(\kappa)$ to contain a timed word $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$ such that $\tau_j - \tau_i \in I$ (where i is the anchor point of κ). Just consider an example where both σ'_{j-1} and σ'_{j+1} contain I but σ'_j does not. Hence, (i) fails to hold. Intuitively, this is because the intervals in Interval words are only positive witnesses for their timing constraints. That is, presence of an interval I implies the timing constraint corresponding to I but absence of it does not imply negation of the timing constraint.

Induction: The induction case is trivial, as both the modalities of TPTL and LTL are identical with exactly the same semantics. For the sake of completeness, we enumerate this trivial argument. Let γ be any arbitrary formulae such that lemma holds for every subformulae of γ [Induction Hypothesis]. We now show that the above lemma holds γ, too.

- *Case 1*: Suppose $\gamma = \gamma_1 \wedge \gamma_2$.
  Proving (i): For any $\kappa, j \models \mathrm{LTL}(\gamma) \Rightarrow \kappa, j \models \mathrm{LTL}(\gamma_1) \wedge \kappa, j \models \mathrm{LTL}(\gamma_2)$. This along with the induction hypothesis (i.e., the lemma holds for γ_1 and γ_2) implies, $\forall \rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n).\rho, i \in \mathsf{Time}(\kappa) \rightarrow \rho, j, [x =: \tau_i] \models \gamma_1$ and $\forall \rho' = (\sigma'_1, \tau'_1) \dots (\sigma'_n, \tau'_n).i \in \mathsf{Time}(\kappa) \rightarrow \rho', j, [x =: \tau'_i] \models \gamma_2$. Which is equivalent to $\forall \rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n).\rho, i \in \mathsf{Time}(\kappa) \rightarrow \rho, j, [x =: \tau_i] \models \gamma$. Hence, (i) holds for $\gamma = \gamma_1 \wedge \gamma_2$.
  Proving (ii): Let $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$ be any arbitrary timed word and let $i, j \in dom(\rho)$ be some arbitrary pair of points in ρ. Then, $\rho, j, [x =: \tau_i] \models \gamma \Rightarrow \rho, j, [x =: \tau_i] \models \gamma_1 \wedge \rho, j, [x =: \tau_i] \models \gamma_2$. This along with the induction hypothesis (i.e., (ii) holds for γ_1 and γ_2) implies for $\kappa = \mathsf{Can}(I_\nu, \rho, i)$, $\kappa, j \models \mathrm{LTL}(\gamma_1) \wedge \kappa, j \models \mathrm{LTL}(\gamma_2)$. Hence, $\kappa, j \models \mathrm{LTL}(\gamma)$.
- *Case 2*: Suppose $\gamma = \gamma_1 \vee \gamma_2$.
  Proving (i): For any $\kappa, j \models \mathrm{LTL}(\gamma)$, $\kappa, j \models \mathrm{LTL}(\gamma_1)$, or $\kappa, j \models \mathrm{LTL}(\gamma_2)$. If $\kappa \models \mathrm{LTL}(\gamma_1)$. Then, every timed $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$ where $\rho, i \in \mathsf{Time}(\kappa)$ is s.t. $\rho, j, [x =: \tau_i] \models \gamma_1$ (and hence γ) because (i) holds for γ_1 by induction hypothesis. Similarly, if $\kappa \models \mathrm{LTL}(\gamma_2)$. Then, every timed $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$ where $\rho, i \in \mathsf{Time}(\kappa)$ is s.t. $\rho, j, [x =: \tau_i] \models \gamma_2$ (and hence γ) because (i) holds for γ_2 (again by induction hypothesis). Hence, (i) holds for γ, too.
  Proving (ii): Suppose (ii) does not hold for γ. This implies there exists a timed word $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$ such that $\rho, j, [x =: \tau_i] \models \gamma_1 \vee \gamma_2$ for some $i \in dom(\rho)$ but for $\kappa = \mathsf{Can}(I_\nu, \rho, i)$, $\kappa, j \not\models \mathrm{LTL}(\gamma_1)$ and $\kappa, j \not\models \mathrm{LTL}(\gamma_2)$. This contradicts the induction hypothesis.

- *Case 3*: Suppose $\gamma = \gamma_1 \mathsf{U} \gamma_2$.
  Proving (i): By semantics of U, for any $\kappa, j \models \mathrm{LTL}(\gamma)$ implies (a) $\exists j' > j$ such that $\kappa, j' \models \mathrm{LTL}(\gamma_2)$ and $\forall j < j'' < j'. \kappa, j'' \models \mathrm{LTL}(\gamma_1)$.[8] As (i) holds for γ_1 and γ_2 by induction hypothesis, (a) implies that for any word $\rho, i \in \mathsf{Time}(\kappa)$, (b)$\exists j' > j$ such that $\rho, j', [x =: \tau_i] \models \gamma_2$ and $\forall j < j'' < j. \rho, j'', [x =: \tau_i] \models \gamma_1$. Note that (b) iff $\rho, j, [x =: \tau_i] \models \gamma$. Hence, (i) holds for γ.
  Proving (ii): Let $\rho = (\sigma_1, \tau_1) \dots (\sigma_n, \tau_n)$ be any arbitrary timed word and let $i, j \in dom(\rho)$ be some arbitrary time points of ρ. Then, $\rho, j, [x =: \tau_i] \models \gamma$, implies that there exists a point $j' > j$ such that (c) $\rho, j', [x =: \tau_i] \models \gamma_2$ and (d) for all $j < j'' < j'$ $\rho, j'', [x =: \tau_i] \models \gamma_1$. Let $\kappa = \mathsf{Can}(I_\nu, \rho, i)$. By induction hypothesis, (c) implies there exists a point $j' > j$ such that $\kappa, j' \models \mathrm{LTL}(\gamma_2)$ and (d) implies for all $j < j'' < j'$ $\kappa, j'' \models \mathrm{LTL}(\gamma_1)$. Hence, $\kappa, j \models \mathrm{LTL}(\gamma)$.
- *Case 4*: $\gamma = \gamma_1 \mathsf{S} \gamma_2$. This case is symmetric to Case 3 and can be argued similarly.
- *Case 5*: $\gamma = \mathcal{G}(\gamma')$.
  Proving (i): $\kappa, j \models \mathrm{LTL}(\gamma)$ iff $\forall j' > j. \kappa, j' \models \mathrm{LTL}(\gamma')$. By induction hypothesis, the lemma statement holds for γ'. Hence, for every $\rho, i \in \mathsf{Time}(\kappa)$, $\forall j' > j. \rho, j', [x =: \tau_i] \models \gamma'$. Hence, $\rho, j, [x =: \tau_i] \models \gamma$.
  Proving (ii): $\rho, j, [x =: \tau_i] \models \gamma$. This implies $\forall j' > j. \rho, j', [x =: \tau_i] \models \gamma'$. By induction hypothesis, if $\kappa = \mathsf{Can}(\mathcal{I}_\nu, \rho, i)$, then $\forall j' > j. \kappa, j' \models \mathrm{LTL}(\gamma')$. Hence, $\kappa, j \models \mathrm{LTL}(\gamma)$.
- *Case 6*: $\gamma = \mathcal{H}(\gamma')$. This case is symmetric to Case 5 and can be argued similarly. □

5.1.2 LTL to NFA over Collapsed Interval Words. It is known that for any LTL[U, S] formula, one can construct an equivalent NFA with at most exponential number of states [20]. We reduce the LTL formula α to an equivalent NFA $A_\alpha = (Q, \mathsf{init}, 2^{\Sigma'}, \delta', F)$ over I_ν-interval words, where $\Sigma' = 2^{\Sigma \cup \mathcal{I}_\nu \cup \{\mathsf{anch}\}}$.

Example 5.7. Consider the LTL formula α from Example 5.1, Figure 4, is the automaton equivalent to α. Note that we constructed this automaton without using the procedure in Reference [20], as α was not very complicated. But, in general, we need to rely on the procedure mentioned in Reference [20]. Moreover, Figure 5 is the collapsed automaton, A constructed from automaton A_α in Figure 4.

From A_α, we construct an automaton $A = (Q, \mathsf{init}, 2^{\Sigma'}, \delta, F)$ s.t. $L(A) = \mathsf{Col}(L(A_\alpha))$. Automaton A is obtained from A_α by replacing the set of intervals $\mathcal{I}$ on the transitions by the single interval $\bigcap \mathcal{I}$. In case $\exists I_1, I_2 \in \mathcal{I}$ s.t. $I_1 \cap I_2 = \emptyset$ (i.e., with contradictory interval constraints), the transition is omitted in A. Also, note that anch semantically implies interval $[0, 0]$. Hence, all the intervals that contain $[0, 0]$ along with the proposition anch are omitted from the transition labels, as those intervals enforce redundant constraints (constraints that are already enforced by anch). Moreover, if any transition label contains an interval I disjoint from $[0, 0]$ appearing along with the proposition anch, then the transition is omitted, as the presence of anch and I at any point j implies contradictory timing constraints on j. Note that each transition of the collapsed automaton is labelled by letters of the form S or $S \cup \{\mathsf{anch}\}$ or $S \cup \{I\}$ where $S \subseteq \Sigma$ and $I \in I_\nu = CL(\mathcal{I}_\nu)$. This gives $L(A) = \mathsf{Col}(L(A_\alpha))$. This implies $\mathsf{Time}(L(A)) = \mathsf{Time}(L(A_\alpha)) = \mathsf{Time}(L(\alpha)) = L_{pt}(\psi)$. Hence, from this point onwards, we have language of collapsed I_ν (rather than $\mathcal{I}_\nu$) interval words capturing the semantics of the given TPTL formula, ψ.

Example 5.8. Figure 5 is the collapsed automaton, A constructed from automaton A_α in Figure 4, as mentioned above.

In the upcoming Sections 5.2 and 5.3, we show that we can construct a PnEMTL formula ϕ using intervals in I_ν such that it accepts all the pointed timed words in $\mathsf{Time}(L(A))$. In general,

[8] Note that, $\mathrm{LTL}(\varphi_1 \cup \varphi_2) = \mathrm{LTL}(\varphi_1) \cup \mathrm{LTL}(\varphi_2)$.

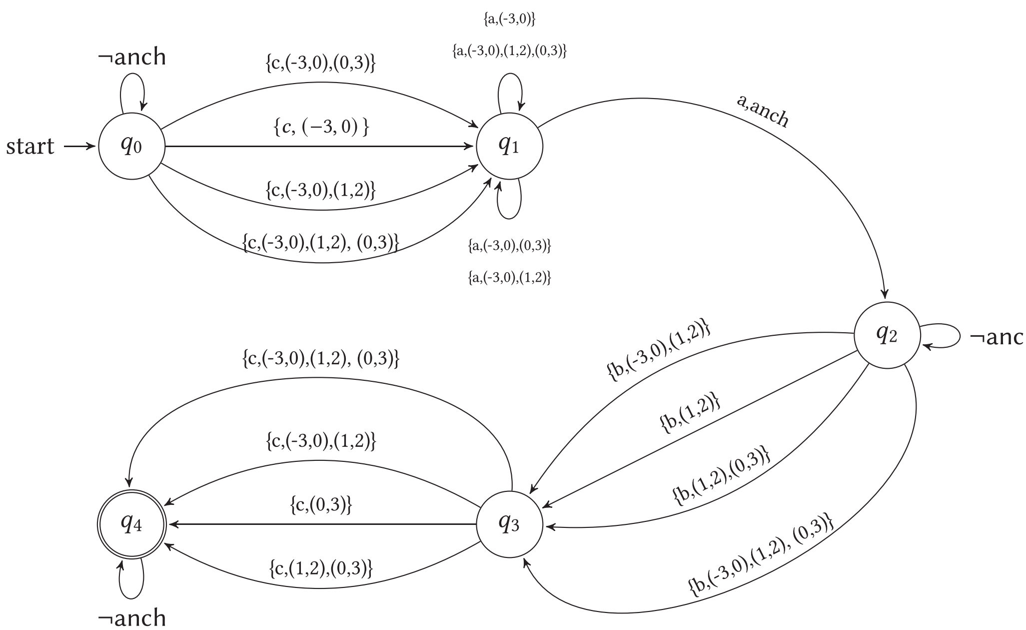

Fig. 4. The automaton, A_α, above is equivalent to LTL formula α from Example 5.1. For the sake of succinctness, $q_1 \overset{a,anch}{\leftarrow} q_2$ denotes set of all transitions from q_1 to q_2 labelled by some subset of $S = \{\text{anch}, a, b, c, (-3,0), (1,2), (0,3)\}$ containing a and anch, and not containing b and c. Similarly, transition labelled ¬anch denotes set of all the transitions labelled by some subset of S contains either a or b or c exclusively.

the construction of PnEMTL formula from the NFA over collapsed interval words along with the construction of NFA over collapsed interval words from NFA over interval words (construction of A from A_α) proves the following result:

Theorem 5.9. *Let $L(A)$ be the language of any I_ν-interval words definable by any NFA A. We can construct a PnEMTL formula ϕ s.t. $\rho, i \models \phi$ iff $\rho, i \in \mathsf{Time}(L(A))$. Moreover, the number of distinct modalities is at most $|A|$, Number of Boolean operators is in $O2^{Poly(|A|)}$ and arity of ϕ is at most $2|I_\nu|^2 + 1$.*

5.2 Constructing Normalized Automata for Each Type Sequence

In this section, we elaborate on step 2 of the reduction. We discuss here how to partition W, the set of all collapsed I_ν-interval words, into finitely many classes. Each class is characterized by its **type** given as a finite sequences seq over $I_\nu \cup \{\text{anch}\}$. For any collapsed $w \in W$, its type seq gives an ordering between $\mathsf{anch}(w)$, $\mathsf{first}(w, I)$ and $\mathsf{last}(w, I)$ for all $I \in I_\nu$, such that, any $I \in I_\nu$ appears at most twice and anch appears exactly once in seq. For instance, $\text{seq} = I_1 I_1 \text{anch} I_2 I_2$ is a sequence different from $\text{seq}' = I_1 I_2 \text{anch} I_2 I_1$, since the relative orderings between the first and last occurrences of I_1, I_2 and anch differ in both. Let the set of types $\mathcal{T}(I_\nu)$ be the set of all such sequences; by definition, $\mathcal{T}(I_\nu)$ is finite.

Intuition: For every type $\text{seq} \in \mathcal{T}(I_\nu)$, we construct an automaton A_{seq} that accepts the normalization of all the words of type seq accepted by A (i.e., $L(A_{\text{seq}}) = \{Norm(w) | w \in L(A) \wedge w$ is of type seq$\}$. Hence, $\bigcup_{\text{seq} \in \mathcal{T}(I_\nu)} \mathsf{Time}(L(A_{\text{seq}})) = \mathsf{Time}(\mathsf{Norm}(L(A))) = \mathsf{Time}(L(A))$. Hence, the union of all these newly constructed automata encodes the required timed languages. The

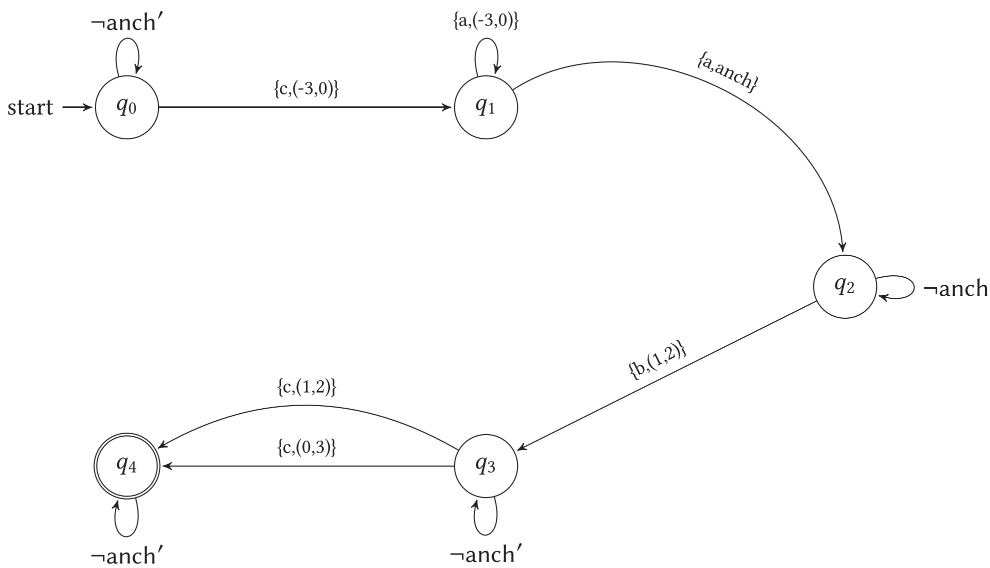

Fig. 5. The automaton, A, is collapsed version of A_α in Example 5.7. Intuitively, we take intersection of all the intervals appearing in a label of a particular transition. In case the intersection is empty, we delete the transition. Following this, if anch appears along with an interval that does not contain $[0, 0]$, then that transition is deleted, as the constraint enforced by that interval is in contradiction with that enforced by anch. Otherwise, the transition is retained by removing the interval from the label, as the constraint enforced by the interval is already enforced by anch. Note that now ¬anch′ denotes set of all the transitions labelled by some subset of S containing either a or b or c exclusively. Moreover, the labels of these transitions contain at most 1 interval.

motivation behind construction of such an automaton is as follows: Each of the words accepted by A_{seq} for any seq $\in \mathcal{T}(I_\nu)$ has bounded number of (as $|\mathsf{seq}| \leq 2 \times |I_\nu| + 1$) time-restricted points. The main reason to do this is so we get automata (i.e., A_{seq}) with structure similar to that shown in Figure 7. Such an automaton over I_ν-interval words can be factored at time-restricted points and its corresponding timed language can then be expressed using a PnEMTL formula, ϕ_{seq}, with arity bounded by the length of sequence seq. The construction of the required formula will be presented in Section 5.3. Hence, restricting to only normalized words makes it possible to construct a PnEMTL formula with bounded arity. Moreover, as $\mathcal{T}(I_\nu)$ is bounded, we can get a bounded size formula, $\phi = \mathsf{Time}(L(A))$, by disjuncting ϕ_{seq} over all possible values of seq $\in \mathcal{T}(I_\nu)$.

Given $w \in W$, let $\mathsf{Boundary}(w) = \{i_1, i_2, \ldots, i_k\}$ be the positions of w that are either $\mathsf{first}(w, I)$ or $\mathsf{last}(w, I)$ for some $I \in I_\nu$ or is $\mathsf{anch}(w)$. Let $w \downarrow_{\mathsf{Boundary}(w)}$ be the subword of w obtained by projecting w to the positions in $\mathsf{Boundary}(w)$, restricted to the subalphabet $2^{I_\nu} \cup \{\mathsf{anch}\}$. For example, $w = \{a, I_1\}\{b, I_1\}\{c, I_2\}\{\mathsf{anch}, a\}\{b, I_1\}\{b, I_2\}\{c, I_2\}$ gives $w \downarrow_{\mathsf{Boundary}(w)}$ as $I_1 I_2 \mathsf{anch} I_1 I_2$. Then, w is in the partition W_{seq} iff $w \downarrow_{\mathsf{Boundary}(w)} =$ seq. Clearly, $W = \bigcup_{\mathsf{seq} \in \mathcal{T}(I_\nu)} W_{\mathsf{seq}}$. Continuing with the example above, w is a collapsed $\{I_1, I_2\}$-interval word over $\{a, b, c\}$, with $\mathsf{Boundary}(w) = \{1, 3, 4, 5, 7\}$, and $w \in W_{\mathsf{seq}}$ for seq $= I_1 I_2 \mathsf{anch} I_1 I_2$, while $w \notin W_{\mathsf{seq}'}$ for seq$' = I_1 I_1 \mathsf{anch} I_2 I_2$.

For type sequence seq $= I_1, I_2, \ldots, I_k$, let $Support(\mathsf{seq})$ give the set of intervals (including anch) occurring in seq. Each such interval occurs 1 or 2 times. Let $Idx(\mathsf{seq}) = \{1 \ldots k + 1\}$. We define function $Status(\mathsf{seq}) : Idx(\mathsf{seq}) \rightarrow Support(\mathsf{seq}) \rightarrow \{pre, mid, post\}$ as follows: Let $j \in Idx(\mathsf{seq})$ and $I \in Support(\mathsf{seq})$. Then, $Status(j)(I) = pre$ if I does not occur in seq strictly before index j. Also, $Status(j)(I) = post$ if I does not occur in seq at or after index j. Finally, $Status(j)(I) = mid$

if I occurs in seq both strictly before j and also at or after index j. For example, for seq = anch $I_1I_2I_1$, we have $Status(2)(I_1) = pre$ and $Status(2)(I_2) = pre$ and $Status(2)(\text{anch}) = post$. Also, $Status(4)(I_1) = mid$ and $Status(4)(I_2) = post$.

Let seq be any sequence in $\mathcal{T}(I_\nu)$. We construct an NFA, Aut_{seq}, which recognizes exactly the collapsed interval words, W_{seq}, of type seq. Automaton $Aut_{\text{seq}} = (Idx(\text{seq})), 1, 2^{\Sigma'}, \delta_2, \{|\text{seq}| + 1\})$. Its transitions are as follows: Let $S \subseteq \Sigma$, $j \in Idx(seq)$, I_j be the jth element of seq and $I \in Support(seq)$. Then,

- $\delta_2(j, S) = j$. Call such transitions as *unconstrained* type transitions.
- $\delta_2(j, S \cup I_j) = j + 1$ if $Status(j)(I_j) = pre$. Note that if I_j occurs exactly once in seq, then the status changes from *pre* to *post* after the transition, and if it occurs twice, the status changes from *pre* to *mid* after the transition. Call such transitions as *progress* transitions (since j increments).
- If $Status(j)(I_j) = mid$, then we have a non-deterministic choice of two transitions. Choice 1: $j + 1 \in \delta_2(j, S \cup I_j)$. In this case, the status of I_j changes from *mid* to *post*. Call this also as *progress* type transition. It corresponds to accepting the second occurrence of I_j. Choice 2: $j \in \delta_2(j, S \cup I_j)$. Call this transition as middle type of transition. This corresponds to accepting a redundant middle occurrence of I_j between its first and last occurrence.
- For $I \neq I_j$ if $Status(j)(I) = mid$, then $\delta_2(j, S \cup I) = j$. This also represents a middle type of transition where redundant middle occurrence of I is accepted. The position j in seq is unchanged and the status of I remains mid after the transition.
- Aut_{seq} has no transitions other than given above.

The following proposition follows directly from the construction:

Proposition 5.10. $L(Aut_{\text{seq}}) = W_{\text{seq}}$.

Given collapsed interval word automaton $A = (Q, \text{init}, 2^{\Sigma'}, \delta, F)$ for the LTL formula as constructed above, the product automaton $A_{prod} = (A \times Aut_{\text{seq}})$ has the property $L(A_{prod}) = (L(A) \cap W_{\text{seq}})$. Thus, A_{prod} accepts the collapsed words belonging to the partition W_{seq} and accepted by A. The automaton A_{prod} has the form $((Q \times Idx(\text{seq})), (\text{init}, 1), 2^{\Sigma'}, \delta_1, F \times \{|\text{seq}| + 1\})$, where δ_1 is obtained by synchronous composition of δ and δ_2 as usual. Observe that in an accepting run of A_{prod} on a word w, the progress transitions increment the index component j of the product state (q, j). These transitions occur exactly at $Boundary(w)$ positions in the word and they represent intervals in w that are retained in the normalized version of w. The middle type transitions, which leave the index component j unchanged, correspond to redundant middle intervals that do not occur in the normalized version of w.

To obtain the automaton A_{seq} accepting normalized words corresponding to words accepted by the product A_{prod}, we project out the redundant intervals in middle type transition in the automaton A_{prod}. Thus, A_{seq} has same states (including initial and final states) but its transition function δ_{seq} differs from the transition function δ_1 of A_{prod}. Let $A_{\text{seq}} = ((Q \times Pos(\text{seq})), (\text{init}, 1), 2^{\Sigma'}, \delta_{\text{seq}}, F \times \{|\text{seq}| + 1\})$. Its transitions are:

- $\delta_{\text{seq}}((q, j), S) = \delta_1((q, j), S$. Thus, unconstrained transitions are identical to A_{prod}.
- If $\delta_1((q_1, j), S \cup \{I\}) = (q_2, j + 1)$, then $\delta_{\text{seq}}((q_1, j), S \cup \{I\}) = (q_2, j + 1)$. Thus, progress transitions are identical to A_{prod}.
- If $\delta_1((q_1, j), S \cup \{I\}) = (q_2, j)$, then $\delta_{\text{seq}}((q_1, j), S) = (q_2, j)$. Thus, redundant interval in middle transitions of A_{prod} are projected out.

The reader may notice the following features of A_{seq}: Let I be any element of where $I_\nu \cup \{\text{anch}\}$. The only transitions with labels of the form $S \cup \{I\}$ (these are called time-constrained transitions)

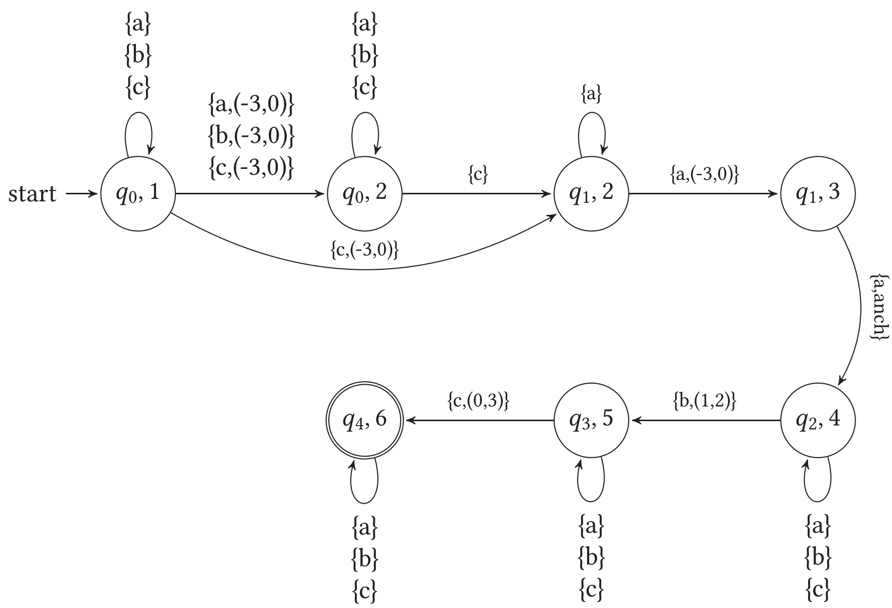

Fig. 6. The automaton A_{seq} depicts the construction of A_{seq} from A (Example 5.8, Figure 5) for seq $=$ $(-3,0)(-3,0)\text{anch}(1,2)(0,3)$. Note that the transition from $q_0, 1$ to $q_0, 2$ is a progress transition. The behavior of $q_0, 2$ is identical to q_0 of A on reading {a} and {b}. On reading {c}, it either behaves like q_0 on c or like q_0 on $\{c, (-3,0)\}$ as the $Status(1, (-3,0)) = mid$.

are the progress transition, and they occur in order specified by seq in any accepting run. All other transitions are labelled with $S \subseteq \Sigma$. They are unconstrained. Hence, the automaton graph partitions into disjoint subgraphs with only unconstrained transitions. These subgraphs are connected by progress transitions. See Figure 7.

Example 5.11. Given automata A from Figure 5 in Example 5.8, we construct $\mathcal{A}_{\text{seq}}$ for different type of sequences accepted by A. We illustrate the construction of A_{seq} where seq $=$ $(-3,0)(-3,0)\text{anch}(1,2)(1,2)$, in Figure 6.

From the construction of A_{seq}, the following property clearly holds:

Proposition 5.12. $L(A_{\text{seq}}) = Normalize(L(A) \cap W_{\text{seq}})$.

From the above proposition, it follows that $\bigcup_{\text{seq} \in \mathcal{T}(I_\nu)} L(A_{\text{seq}}) = \mathsf{Norm}(L(A))$. Hence, using Theorem 5.2, we get

$$\bigcup_{\text{seq} \in \mathcal{T}(I_\nu)} \mathsf{Time}(L(A_{\text{seq}})) = \mathsf{Time}(\mathsf{Norm}(L(A))) = \mathsf{Time}(L(A)) = L_{pt}(\psi).$$

Hence, A_{seq} is the required Normalized Automata for type seq.

5.3 Reducing NFA of Each Type to PnEMTL

Our next step is to reduce the NFAs A_{seq} corresponding to each type seq as constructed in the previous step to a language equivalent formula of logic PnEMTL. This is step 3 of the reduction. The words in $L(A_{\text{seq}})$ are all normalized and have at most $2|I_\nu| + 1$-time-restricted points. Thanks to this, its corresponding timed language can be expressed using PnEMTL formulae with arity at most $2|I_\nu|$.

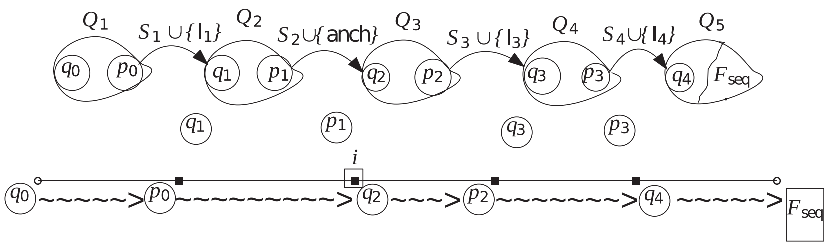

Fig. 7. Figure representing set of runs $A_{\mathsf{I_1 anch I_3 I_4}}$ of type $Qseq$ where each $S_i \subseteq \Sigma$ and each sub-automaton Q_i has only transitions without any intervals. Here, $Qseq = T_1T_2T_3T_4$, for $1 \leq i \leq 4$, $T_i = (p_{i-1} \overset{S_i \cup \{I_i\}}{\to} q_i)$, $I_2 = \{\text{anch}\}$.

For each A_{seq}, we construct PnEMTL formula ϕ_{seq} such that, for a timed word ρ with $i \in dom(\rho)$, $\rho, i \models \phi_{\text{seq}}$ iff $\rho, i \in \mathsf{Time}(L(A_{\text{seq}}))$.

5.3.1 Important Notations. For any NFA, $N = (St, \Sigma, i, Fin, \Delta)$, $q \in Q$ $F' \subseteq Q$, let $N[q, F'] = (St, \Sigma, q, F', \Delta)$. For brevity, we denote $N[q, \{q'\}]$ as $N[q, q']$. We denote by $\mathsf{Rev}(N)$, the NFA N' that accepts the reverse of $L(N)$. The right/left concatenation of $a \in \Sigma$ with $L(N)$ is denoted $N \cdot a$ and $a \cdot N$, respectively.

Lemma 5.13. *We can construct a PnEMTL formula ϕ_{seq} with arity $\leq 2|I_\nu|+1$ and size $O(|A_{\text{seq}}|^{|\text{seq}|})$ containing intervals from I_ν s.t. $\rho, i \models \phi_{\text{seq}}$ iff $\rho, i \in \mathsf{Time}(L(A_{\text{seq}}))$.*

Proof. Let seq $= I_1\ I_2\ \ldots\ I_n$, and $I_j =$ anch for some $1 \leq j \leq n$.

Intuition: Note that we know the sequence, seq, of intervals that we will read. Moreover, this sequence is of bounded size. Hence, any accepting run will pass through at most n transitions, $T_1, T_2, \ldots, T_n$ labelled with some interval or anch. Thus, the part of accepting run between T_i and T_{i+1} for $i \in \{1, \ldots, n-1\}$ contains transitions labelled only by some non-empty subset of Σ. Hence, the set of words read by runs between T_i and T_{i+1} for the set of runs passing through transitions, $T_1, T_2, \ldots, T_n$, can be expressed by an automaton, A_{i+1}, over alphabets in $2^\Sigma \setminus \emptyset$.

Proof: Before starting the proof, notice the structure of A_{seq}. The state space is partitioned in to sets $Q_1, \ldots Q_{n+1}$. Transitions within any partition Q_i are unconstrained transitions. From any state in Q_i there are constrained transition on proposition containing interval I_i that leads to some state in Q_{i+1}. Hence, set of states in Q_i are reachable exactly after $i-1$ time-constrained transitions. Let $\Gamma = 2^\Sigma$ and Qseq $= T_1\ T_2\ \ldots T_n$ be a sequence of time-constrained transitions of A_{seq} where for any $1 \leq i \leq n$, $T_i = p_{i-1} \overset{S'_i}{\to} q_i$, $S'_i = S_i \cup \{I_i\}$, $S_i \subseteq \Sigma$, we define $\mathsf{R_{Qseq}}$ as set of accepting runs containing transitions $T_1\ T_2\ \ldots T_n$. Hence, the runs in $\mathsf{R_{Qseq}}$ are of the following form:

$T_{0,1}\ T_{0,2} \ldots T_{0,m_0}\ T_1\ T_{1,1}\ \ldots T_{1,m_1}\ T_2\ \cdots\cdots\ T_{n-1,1}\ T_{n-1,2} \ldots T_n\ T_{n,1} \ldots T_{n+1}$

where the source of the transition $T_{0,1}$ is q_0 and the target of the transition T_{n+1} is any accepting state of A_{seq}. Moreover, all the transitions $T_{i,j}$ for $0 \leq i \leq n$, $1 \leq j \leq n_i$ are unconstrained transitions of the form $(p' \overset{S_{i,j}}{\to} q')$ where $S_{i,j} \subseteq \Sigma$ and $p', q' \in Q_{i+1}$. Hence, only $T_1, T_2, \ldots T_n$ are labelled by any interval from I_ν. Moreover, only on these transitions the position counter (i.e., second element of the state) increments. Let $\mathsf{A}_i = (Q_i, 2^\Sigma, q_{i-1}, \{p_{i-1}\}, \delta_{\text{seq}}) \equiv A_{\text{seq}}[q_{i-1}, p_{i-1}]$ for $1 \leq i \leq n$ and $\mathsf{A}_{n+1} = (Q_{n+1}, 2^\Sigma, q_n, F_{\text{seq}}, \delta_{\text{seq}}) \equiv A[q_n, F]$. Let $\mathcal{W}_{Qseq}$ be set of words associated with any run in R_{Qseq}. In other words, any word w in $\mathcal{W}_{Qseq}$ admits an accepting run on A_{seq} that starts from q_0 reads letters without intervals (i.e., symbols of the form $S \subseteq \Sigma$) ends up at p_0, reads S'_1, ends up at q_1 reads letters without intervals, ends up at p_1, reads S'_2, and so on. Refer to Figure 7 for illustration. Hence, $w \in W_{Qseq}$ if and only if

$w \in L(\mathsf{A}_1).S'_1.L(\mathsf{A}_2).S'_2.\cdots.L(\mathsf{A}_n).S'_n.L(\mathsf{A}_{n+1})$. Let $\mathsf{A}'_k = S_{k-1} \cdot \mathsf{A}_k \cdot S_k$ for $1 \leq k \leq n+1$, with $S_0 = S_{n+1} = \epsilon$.[9] Let $\rho = (b_1, \tau_1) \ldots (b_m, \tau_m)$ be a timed word over Γ. Then $\rho, i_j \in \mathsf{Time}(W_{Qseq})$ iff $\exists$ $1 \leq i_1 \leq i_2 \leq \cdots \leq i_{j-1} \leq i_j \leq i_{j+1} \leq \cdots \leq i_n \leq m$ s.t. $\bigwedge_{k=1}^{j-1}[(\tau_{i_k} - \tau_{i_j} \in I_k) \wedge \mathsf{Seg}^-(\rho, i_{k+1}, i_k, \Gamma) \in L(\mathsf{Rev}(\mathsf{A}'_k))] \wedge \bigwedge_{k=j}^{n}[(\tau_{i_k} - \tau_{i_j} \in I_k) \wedge \mathsf{Seg}^+(\rho, i_k, i_{k+1}, \Gamma) \in L(\mathsf{A}'_k)]$, where $i_0 = 1$ and $i_{n+1} = m$. Hence, by semantics of $\mathcal{F}^k$ and $\mathcal{P}^k$ modalities, $\rho, i \in \mathsf{Time}(\mathcal{W}_{Qseq})$ if and only if $\rho, i \models \phi_{\mathrm{qseq}}$ where $\phi_{\mathrm{qseq}} = \mathcal{P}^j_{I_{j-1},\ldots,I_1}(\mathsf{Rev}(\mathsf{A}'_1), \ldots, \mathsf{Rev}(\mathsf{A}'_j))(\Gamma) \wedge \mathcal{F}^{n-j}_{I_{j+1},\ldots,I_n}(\mathsf{A}'_{j+1}, \ldots, \mathsf{A}'_{n+1})(\Gamma)$. Let State$-$seq be the set of all possible sequences of the form Qseq. As A_{seq} accepts only words that have exactly n time-restricted points, the number of possible sequences of the form Qseq is bounded by $|Q|^n$. Hence, any word $\rho, i \in \mathsf{Time}(L(A_{\mathrm{seq}}))$ iff $\rho, i \models \phi_{\mathrm{seq}}$ where $\phi_{\mathrm{seq}} = \bigvee_{\mathrm{qseq} \in \mathrm{State-seq}} \phi_{\mathrm{qseq}}$. Disjuncting over all possible sequences seq $\in \mathcal{T}(I_\nu)$, we get the required formula $\phi = \bigvee_{\mathrm{seq} \in \mathcal{T}(I_\nu)} \phi_{\mathrm{seq}}$.

Example 5.14. As a continuation of our running example, we give a construction of PnEMTL formula ϕ_{seq} for automaton A_{seq} from Example 5.11, Figure 6. Note that the accepting runs of the automaton A_{seq} can either contain transition $q_0, 1 \to q_2, 2$ bypassing $q_1, 2$ or pass via $q_1, 2$. The timed behaviors for the former (and latter) case can be captured by formula ϕ_1 (and ϕ_2, respectively), where $\phi_1 = a \wedge \phi_{fut} \wedge \phi_{past,1}$ and $\phi_2 = a \wedge \phi_{fut} \wedge \phi_{past,1}$ where,

$$\phi_{fut} = \mathcal{F}^2_{(1,2),(0,3)}(a.\Sigma^*.b, b.\Sigma^*.c, c.\Sigma^*)(\Sigma),$$
$$\phi_{past,1} = \mathcal{P}^2_{(0,3),(0,3)}(a.a, a.a^*.c, c.\Sigma^*, \Sigma^*)(\Sigma),$$
$$\phi_{past_2} = \bigvee_{x \in \{a,b,c\}} \mathcal{P}^2_{(0,3),(0,3)}(a.a, a.a^*.c.\Sigma^*.x, x.\Sigma^*)(\Sigma).$$

The a in blue is the a occurring at the present position (i.e., a that occurred along with the anchor point in the interval word automata $\mathcal{A}_{\mathrm{seq}}$). Moreover, $\phi_{\mathrm{seq}} = \phi_1 \vee \phi_2$. □

The construction in Section 5.2, Proposition 5.12, and proof of Lemma 5.13 imply Theorem 5.9. Note that, if ψ is a simple 1-TPTL formula with intervals in $\mathcal{I}_\nu$, then the equivalent PnEMTL formula, ϕ, constructed above contains only interval in $\mathsf{CL}(\mathcal{I}_\nu)$. Hence, we have the following theorem:

Theorem 5.15. *For a simple non-adjacent 1-TPTL formula ψ containing intervals from $\mathcal{I}_\nu$, we can construct a non-adjacent PnEMTL formula ϕ, s.t. for any valuation v, $\rho, i, v \models \psi$ iff $\rho, i \models \phi$ where, $|\phi| = O(2^{Poly(|\psi|)})$ and arity of ϕ is at most $2|\mathcal{I}_\nu|^2 + 1$.*

Proof. Let $|\psi| = m$, $|\mathcal{I}_\nu| = n$.

- Construct an LTL formula α over interval words such that $\rho, i \models \varphi$ if and only if $\rho, i \models \mathsf{Time}(L(\alpha))$ as in Section 5.1.1 such that $|\alpha| = O(n)$.
- Reduce the LTL formula α to language equivalent NFA A' using Reference [20]. This has the complexity $O(2^n)$. This step is followed by reducing A' to A over interval words over I_ν such that $L(A) = \mathsf{Col}(L(A'))$. Note that $|I_\nu| = |\mathcal{I}_\nu|^2 = O(n^2)$ Section 5.1.2.
- As shown in Section 5.2, for any type seq, we can construct A_{seq} from A such that $L(A_{\mathrm{seq}}) = \mathsf{Norm}(L(A_{\mathrm{seq}}) \cup W_{\mathrm{seq}})$ with number of states $k = O(2^{Poly(m)})$.
- As shown in Section 5.3, for any seq, we can construct ϕ_{seq} using intervals from I_ν such that $\rho, i \models \phi_{\mathrm{seq}}$ iff $\rho, i \in L(A_{\mathrm{seq}})$. Note that $Time(L(\varphi)) = Time(L(A)) = \bigcup_{\mathrm{seq} \in \mathcal{T}(I_\nu)} Time(L(A_{\mathrm{seq}}))$. Note that $|\mathcal{T}(I_\nu)| \leq (n)^{2n^2} = O(2^{Poly(n)})$. Size of formula ϕ_{seq} is $(2^{n*m}) \leq 2^{m^2}$. Moreover, the arity of the formula $\phi_{\mathrm{seq}} = 2 \times |\mathrm{seq}| = O(2 \times |I_\nu| + 1)$ (as each interval from I_ν appears at most twice in seq, and anch appears exactly once) $= O(n^2)$. Hence, $\rho, i \models \psi$ if and only if $\rho, i \models \phi$ where $\phi = \bigvee_{\mathrm{seq} \in \mathcal{T}(I_\nu)} \phi_{\mathrm{seq}}$ and the timing intervals used in ϕ comes from I_ν. Note

[9] We mention $\mathsf{A}'_k = S_{k-1} \cdot \mathsf{A}_k \cdot S_k$ instead of $\cdot \mathsf{A}_k \cdot S_k$ due to the non-strict inequalities in the semantics of PnEMTL modalities.

that if $\mathcal{I}$ is non-adjacent, then I_ν is non-adjacent, too. Hence, we get a non-adjacent PnEMTL formula ϕ the size of which is $O(2^{Poly(m)})$ and arity is $O(n^2)$. □

The above theorem (Theorem 5.15) is lifted to a general (non-simple) 1-TPTL formula ψ as follows: Given a 1-TPTL formula ψ in DAG form, we first convert innermost simple sub-formulae ζ_i^1 to their equivalent PnEMTL formulae $\hat{\zeta}_i^1$. We substitute a fresh witness proposition a_i^1 in place of ζ_i^1 giving formula $\psi^1 = \psi[a_i^1/\zeta_i^1]$. Superscript 1 states that we have eliminated depth 1 simple subformulae. We repeat the procedure for ψ^1 giving ψ^2 where we introduce depth 2 witness propositions a_i^2 for depth 1 simple sub-formula ζ_i^2 in ψ^1. We recursively apply this procedure till a purely propositional formula ψ^k is obtained having Σ as well as witness variables. We substitute top depth witness variable a_i^k by equivalent PnEMTL formulae $\hat{\zeta}_i^k$. This formula refers to lower-level witness variables of the form a_j^l with $l < k$. We recursively substitute these witness variables by their equivalent PnEMTL formulae $\hat{\zeta}_j^l$, keeping the formula in DAG form. This process is repeated till we obtain a pure PnEMTL formula $\hat{\psi}$ without witness proposition, which is equivalent to ψ. Thus, we have the following result:

Theorem 5.16. *Any (non-adjacent) 1-TPTL formula ψ with intervals in $\mathcal{I}_\nu$ can be reduced to an equivalent (non-adjacent) PnEMTL, ϕ, with $|\phi| = 2^{Poly(|\psi|)}$ and arity of $\phi = O(|\mathcal{I}_\nu|^2)$ such that $\rho, i \models \psi$ iff $\rho, i \models \phi$.*

6 MSO WITH GUARDED METRIC QUANTIFIERS, GQMSO

In this section, we define an extension of **MSO[<] with Guarded Metric Quantifiers (GQMSO)**. The logic is a natural extension of QMLO and Q2MLO of Hirshfeld and Rabinovich where a single metric quantifier is generalized to an anchored block of metric quantifiers of arbitrary depth. We show that PnEMTL is expressively complete for this logic. We define non-adjacency restriction in context of GQMSO and show that the non-adjacency is preserved while translating from PnEMTL to GQMSO and vice versa. Hence, the reduction (from GQMSO to PnEMTL) also serves as a proof of decidability for satisfiability checking of Non-adjacent GQMSO. This is by far the most general fragment of MSO[<, +ℕ] (syntactically) for which satisfiability checking is decidable. As a corollary, we get that (non-adjacent) 1-TPTL is expressively complete for (non-adjacent) GQFO, the first-order fragment of (non-adjacent) GQMSO.

6.1 GQMSO: Syntax and Semantics

We define a real-time logic GQMSO that is interpreted over timed words. It includes MSO[<] over words with respect to some alphabet Σ. This is extended with a notion of time-constraint formula $\psi(t)$, where t is a free first-order variable. All variables in our logic range over positions in the timed word and not over timestamps (unlike continuous interpretation of these logics). There are two sorts of formulae in GQMSO that are mutually recursively defined: $\mathsf{MSO}^{\mathsf{UT}}$ and $\mathsf{MSO}^{\mathsf{T}}$ (where UT stands for untimed and T for timed). An $\mathsf{MSO}^{\mathsf{UT}}$ formula ϕ has no real-time constraints except for the time-constraint subformula $\psi(t) \in \mathsf{MSO}^{\mathsf{T}}$. A formula $\psi(t)$ has only one free variable t (called anchor), which is a first-order variable. $\psi(t)$ is defined as a block of real-time-constrained quantification applied to a GQMSO formula with no free second-order variables; it has the form $\mathcal{Q}_1 t_1 . \mathcal{Q}_2 t_2 . \ldots . \mathcal{Q}_j t_j .\ \phi(t, t_1, \ldots t_j)$ where $\phi \in \mathsf{MSO}^{\mathsf{UT}}$. All the metric quantifiers in the quantifier block constrain their variable relative only to the anchor t. The precise syntax follows below.[10]

[10]In Reference [33], a similar logic called QkMSO was defined. QkMSO had yet another restriction: It can only quantify positions strictly in the future, and hence was not able to express past timed specifications.

Remark: This form of real time constraints in first-order logic were pioneered by Hirshfeld and Rabinovich [25] in their logic Q2MLO (with only non-punctual guards) and its punctual extension was later shown to be expressively complete to FO[<, +1] by Hunter [28] over signals. Here, we extend the quantification to an **anchored block of quantifiers** of arbitrary depth.

We have a two sorted logic consisting of MSO^{UT} formulae ϕ and time-constrained formulae ψ. Let $a \in \Sigma$, and let t, t' range over first-order variables, while T range over second-order variables. The syntax of $\phi \in \mathrm{MSO}^{\mathrm{UT}}$ is given by:

$t = t' \mid t < t' \mid Q_a(t) \mid T(t) \mid \phi \wedge \phi \mid \neg\phi \mid \exists t.\phi \mid \exists T\phi \mid \psi(t)$.

Here, $\psi(t) \in \mathrm{MSO}^{\mathrm{T}}$ is a time-constraint formula whose syntax and semantics are given a little later. A formula in $\mathrm{MSO}^{\mathrm{UT}}$ with first-order free variables $t_0, t_1, \ldots t_k$ and second-order free variables $T_1, \ldots, T_m$ is denoted $\phi(t_0, \ldots t_k, T_1, \ldots, T_m)$. The semantics of such formulae is as usual. Let $\rho = (\sigma_1, \tau_1) \ldots (\sigma_n, \tau_n)$ be a timed word over Σ. Given ρ, positions $i_0, \ldots, i_k$ in $dom(\rho)$, and sets of positions $A_1, \ldots, A_m$ with $A_i \subseteq dom(\rho)$, we define $\rho, (i_0, i_1, \ldots, i_k, A_1, \ldots, A_m) \models \phi(t_0, t_1, \ldots t_k, T_1, \ldots, T_m)$ inductively in MSO[<].

- $(\rho, i_0, \ldots, i_k, A_1, \ldots, A_m) \models t_x < t_y$ iff $i_x < i_y$,
- $(\rho, i_0, \ldots, i_k, A_1, \ldots, A_m) \models Q_a(t_x)$ iff $a \in \sigma_{i_x}$,
- $(\rho, i_0, \ldots, i_k, A_1, \ldots, A_m) \models T_j(t_x)$ iff $i_x \in A_j$,
- $(\rho, i_0, \ldots, i_k, A_1, \ldots, A_m) \models \exists t'.\phi(t_0, \ldots t_k, t', T_1, \ldots, T_m)$ iff $(\rho, i_0, \ldots, i_k, i', A_1, \ldots, A_m) \models \phi(t_0, \ldots t_k, t', T_1, \ldots, T_m)$ for some $i' \in dom(\rho)$.

The **time-constraint** formula $\psi(t) \in \mathrm{MSO}^{\mathrm{T}}$ has the form: $\mathcal{Q}_1 t_1.\mathcal{Q}_2 t_2. \ldots \mathcal{Q}_j t_j.\ \phi(t, t_1, \ldots t_j)$ where $t_1, \ldots, t_j$ are first-order variables and $\phi \in \mathrm{MSO}^{\mathrm{UT}}$. Each quantifier $\mathcal{Q}_x t_x$ has the form $\overline{\exists} t_x \in t + I_x$ or $\overline{\forall} t_x \in t + I_x$ for a time interval $I_x \in \mathcal{I}_{\mathrm{int}}$. $\mathcal{Q}_x$ is called a metric quantifier. Note that each metric quantifier constrains its variable only relative to the anchor variable t. Moreover, $\psi(t)$ has no free second-order variables. The semantics of such an anchored metric quantifier is obtained recursively as follows: Let

$$(\rho, i_0, i_1, \ldots, i_{j-1}) \models \overline{\exists} t_j \in t + I_j.\phi(t, t_1, \ldots, t_j) \text{ iff } \left\{\begin{matrix}\text{there exists } i_j \text{ such that } \tau_{i_j} \in \tau_{i_0} + I_j \text{ and,} \\ (\rho, i_0, i_1 \ldots i_j) \models \phi(t, t_1, \ldots, t_j)\end{matrix}\right\},$$

$$(\rho, i_0, i_1, \ldots, i_{j-1}) \models \overline{\forall} t_j \in t + I_j.\phi(t, t_1, \ldots t_j) \text{ iff } \left\{\begin{matrix}\text{for all } i_j \text{ such that } \tau_{i_j} \in \tau_{i_0} + I_j \text{ implies,} \\ (\rho, i_0, i_1 \ldots i_j) \models \phi(t, t_1, \ldots, t_j)\end{matrix}\right\}.$$

Note that metric quantifiers quantify over positions of the timed word, and the metric constraint is applied on the timestamp of the corresponding positions. Each time-constraint formula in GQMSO has exactly one free variable; variables $t_1, \ldots, t_j$ are called time-constrained in $\psi(t)$. If we restrict the grammar of a time-constrained formula $\psi(t) \in \mathrm{MSO}^{\mathrm{T}}$ to contain only a single metric quantifier (i.e., $\mathcal{Q}_1 t_1.\phi(t, t_1)$) and disallow the usage of second-order quantification, then we get the logic **Q2MLO** of Reference [26].

Example 6.1. Consider sequences over $\Sigma = \{a, b\}$ such that the event a is the last event in the first unit interval. $\phi = \exists t.[\{\forall t'.t \leq t'\} \wedge \{\exists s \in t + (0, 1).\forall s' \in t + (0, 1).(s \geq s' \wedge Q_a(s))\}]$.

Example 6.2. Consider sequences over events $\Sigma = \{a, b\}$ such that from every a there was a positive even number of b's in the previous unit interval. $\phi = \forall t.Q_a(t) \rightarrow \psi(t)$ where $\psi(t) = [\overline{\exists} t_f \in t + [-1, 0].\overline{\exists} t_l \in t + [-1, 0]\overline{\forall} t' \in t + [-1, 0].\gamma(t, t_f, t_l, t')$ where $\gamma(t, t_f, t_l, t') = t_f \leq t' \leq t_l \wedge \exists X_o.\exists X_e.X_o(t_f) \wedge X_e(t_l) \wedge \forall t_1.\forall t_2.\ [\{Q_b(t_1) \wedge Q_b(t_2) \wedge \forall t_3.(t_1 < t_3 < t_2 \rightarrow \neg Q_b(t_3))\} \rightarrow \{(X_o(t_1) \wedge \neg X_e(t_1) \wedge X_e(t_2) \wedge \neg X_o(t_2)) \vee (X_e(t_1) \wedge \neg X_e(t_1) \wedge X_o(t_2) \wedge \neg X_o(t_2))\}]$. Here, ϕ is a formula of type $\mathrm{MSO}^{\mathrm{UT}}$ containing the subformula $\psi(t)$ of type MSO^{T}, which in turn contains the formula $\gamma(t, t_f, t_l, t')$ of type $\mathrm{MSO}^{\mathrm{UT}}$.

Note that, while GQMSO extends classical MSO[<], it is not closed under second-order quantification: Arbitrary use of second-order quantification is not allowed, and its syntactic usage, as explained above, is restricted to prevent a second-order free variable from occurring in the scope of the real-time constraint (similar to References [23, 43, 46]). For example, $\exists X.\exists t.[X(t) \wedge \overline{\exists} t' \in t + (1,2) Q_a(t')]$ is a well-formed GQMSO formula, while $\exists X.\exists t.\overline{\exists} t' \in t + (1,2)[Q_a(t') \wedge X(t)]$ is not, since X occurs freely within the scope of the metric quantifier.

Example 6.3. We define a language $\mathsf{L}_{\mathsf{insterr}}$ over the singleton alphabet $\Sigma = \{b\}$ accepting words satisfying the following conditions:

(1) One b with timestamp 0 at the first position. (Positions are counted $1, 2, 3, \ldots$).
(2) Exactly two points in the interval $(0,1)$ at positions 2 and 3 with timestamps called τ_2 and τ_3, respectively.
(3) Exactly one b in $[\tau_2 + 1, \tau_3 + 1]$ at some position p. Other b's can occur freely elsewhere.

The above language was proposed by Lasota and Walukiewicz [34] (Theorem 2.8) as an example of language not recognizable by 1 clock Alternating Timed Automata but expressible by a Deterministic Timed Automata with 2 clocks. Let $S(u,v)$ be the FO[<] formula specifying the successor relation (i.e., $u = v + 1$). This can be specified as the GQMSO formula $\psi = \psi_1 \wedge \psi_3$, where

(1) Let $Pos_1(t) = \neg \exists w.S(t,w)$, $Pos_i(t) = \exists t'.S(t,t') \wedge Pos_{i-1}(t')$. Hence, $Pos_i(t)$ holds only when $t = i$, where $i \in \{1,2,3,4\}$.
(2) Let $\psi_1 = \exists t_1.\ Pos_1(t_1) \wedge (\overline{\exists} t_2 \in t_1 + (0,1).\overline{\exists} t_3 \in t_1 + (0,1).[Pos_2(t_2) \wedge Pos_3(t_3) \wedge \neg \overline{\exists} t \in t_1 + (0,1).Pos_4(t)]$. This states that exactly two positions exist in the initial unit time interval $(0,1)$. Let their timestamps be τ_2 and τ_3.
(3) Let $\psi_2(p) = [\ \overline{\exists} t \in p + [-1,0).Pos_3(t)\ \wedge\ \neg \overline{\exists} t \in p + (-1,0).Pos_2(t)\ ]$. This states that position p lies within $[\tau_2 + 1, \tau_3 + 1]$.
(4) $\psi_3 = \exists p.\ [\psi_2(p) \wedge (\forall q.\psi_2(q) \rightarrow (p = q))]$ states that there is exactly one position satisfying property ψ_2.

Metric Depth. The *metric depth* of a formula φ denoted $(\mathsf{MtD}(\varphi))$ gives the nesting depth of time constraint constructs and is defined inductively: For atomic formulae φ, $\mathsf{MtD}(\varphi) = 0$. $\mathsf{MtD}[\varphi_1 \wedge \varphi_2] = \mathsf{MtD}[\varphi_1 \vee \varphi_2] = max(\mathsf{MtD}[\varphi_1], \mathsf{MtD}[\varphi_2])$ and $\mathsf{MtD}[\exists t.\varphi(t)] = \mathsf{MtD}[\neg\varphi] = \mathsf{MtD}(\varphi(t))$. $\mathsf{MtD}[\mathcal{Q}_1 t_1 \ldots \mathcal{Q}_j t_j \phi] = \mathsf{MtD}[\phi] + 1$. For example, the sentence $\forall t_3\ \overline{\forall} t_1 \in t_3 + (1,2)\ \{Q_a(t_1) \rightarrow (\overline{\exists} t_0 \in t_1 + [1,1]\ Q_b(t_0))\}$ accepts all timed words such that for each a that is at distance $(1,2)$ from some timestamp t, there is a b at distance 1 from it. This sentence has metric depth two with time-constrained variables t_0, t_1.

6.2 GQMSO with Alternation Free Metric Quantifiers (AF-GQMSO)

We define a syntactic fragment of GQMSO, called AF-GQMSO, where all the metric quantifiers in any anchored metric quantifier block only consist existential metric quantifiers. More precisely, AF-GQMSO is a syntactic fragment of GQMSO where the **time constraint** $\psi(t_0)$ has the form $\overline{\exists} t_1 \in t_0 + I_1.\overline{\exists} t_2 \in t_0 + I_2.\ldots.\overline{\exists} t_j \in t_0 + I_j.\ \phi(t_0, t_1, \ldots t_j)$ with $\phi \in \mathsf{MSO}^{\mathsf{UT}}$. Hence, there is no alternation of metric quantifiers within a block of the metric quantifier. Note that the negation of the timed subformula is allowed in the syntax of GQMSO (and hence AF-GQMSO). Hence, alternation free $\overline{\forall}^* \phi$ formulae can also be expressed as equivalent $\neg \overline{\exists}^* \neg \phi$ using AF-GQMSO. We now show that, surprisingly, AF-GQMSO is as expressive as GQMSO.

Theorem 6.4. *The subclass AF-GQMSO is expressively equivalent to GQMSO.*

Proof. Let $\psi(t_0) = Q_1 t_1.Q_2 t_2.\ldots Q_j t_j.\varphi(t_0, \ldots, t_j)$ be any GQMSO formula where every quantifier $Q_i t_i$ is of the form $\overline{\exists} t_i \in t_0 + I_i$ or $\overline{\forall} t_i \in t_0 + I_i$. Let $I_j = [l, u)$ (similar construction can be given for all other type of intervals). We convert the innermost metric quantifier $Q_j t_j$ to a non-metric quantifier by adding (at most) four existential metric quantifiers at the top level of the form $\overline{\exists} t'_j \in t_0 + (-\infty, l).\overline{\exists} t''_j \in t_0 + [u, \infty).\overline{\exists} t_{first,j} \in I_j.\overline{\exists} t_{last,j} \in I_j.$. Intuitively, the variables t'_j will take the value of the last point within interval $(-\infty, l)$ from t_0. Similarly, t''_j will take the value of the first point within interval $[u, \infty)$ from t_0. Moreover, variable $t_{first,j}$ ($t_{last,j}$) will take the value of the first (last, respectively) point within interval I_j from t_0. Hence, we can replace the quantifier $Q_j t_j$ with $\exists t_{first,j} \le t \le t_{last,j}$ if Q_j is an existential metric quantifier and with $\forall t_{first,j} \le t_j \le t_{last,j}$ if Q_j is a universal metric quantifier. Hence, repeating the above steps for $Q_{j-1} \ldots Q_1$, we get a AF-GQMSO with at most $4j$ existential metric quantifiers.

For $1 \le I \le j$, Let φ_i be the subformula $Q_i t_i.Q_2 t_2.\ldots Q_j t_j.\varphi(t_0, \ldots, t_j)$. Other types of intervals can be handled similarly. We eliminate $Q_j t_j$ as follows:

(1) If there is no point within $[l, u)$ of t_0, then the sub- formulae φ_j vacuously evaluates to true if Q_j is a universal metric quantifier and evaluates to false if Q_j is an existential metric quantifier. $C_1 = \neg\overline{\exists} t \in I_j \to Q_1 t_1.Q_2 t_2.\ldots Q_{j-1} t_{j-1}.\gamma$,
where $\gamma_1 = true$ in case Q_j is a universal metric quantifier and $\gamma_1 = false$ otherwise.

(2) If there is a point in $[l, u)$ from t_0, then we add existential metric quantifiers $\overline{\exists} t_{first,j} \in I_j.\overline{\exists} t_{last,j} \in I_j$. (along with some more existential metric quantifiers) at the top level and assert a formula that forces $t_{first,j}$ to be the first point within interval $[l, u)$ from t_0 and $t_{last,j}$ to be the last. Then, the quantifier $\overline{\exists}.t_j \in t_0 + I_j$ ($\overline{\forall}.t_j \in t_0 + I_j$) can be replaced by $\exists.t_{first,j} \le t_j \le t_{last,j}$ ($\forall.t_{first,j} \le t_j \le t_{last,j}$, respectively). Let $\gamma_2 = \forall t_{first,j} \le t_j \le t_{last,j} \varphi(t_0, \ldots, t_j)$ if Q_j is a universal metric quantifier else $\gamma_2 = \exists t_{first,j} \le t_j \le t_{last,j} \varphi(t_0, \ldots, t_j)$ otherwise. Let $S(t, t')$ be the successor predicate that is true iff $' = t + 1$. It is routine to express such a predicate in MSO[<]. Then, $C_2 = \overline{\exists} t \in t_0 + I_j \to C_{2,1} \vee C_{2,2} \vee C_{2,3} \vee C_{2,4}$ where:

- $C_{2,1}$ covers the possibility that there are points that occur within interval (∞, l) and $[u, \infty)$ from t_0. Hence,
$$C_{2,1} = \begin{matrix} \overline{\exists} t'_j \in t_0 + (-\infty, l).\overline{\exists} t''_j \in t_0 + [u, \infty). \\ \overline{\exists} t_{first,j} \in t_0 + [l, u).\overline{\exists} t_{last,j} \in t_0 + [l, u). \\ Q_1 t_1.Q_2 t_2.\ldots Q_{j-1} t_{j-1}. \end{matrix} \begin{Bmatrix} S(t'_j, t_{first,j}) \wedge \\ S(t_{last,j}, t''_j) \wedge \\ \gamma_2 \end{Bmatrix}$$
- $C_{2,2}$ covers the possibility where there is no point occurring within $(-\infty, l)$ but there are points occurring within $[u, \infty)$ from t_0. Hence,
$$C_{2,2} = \begin{matrix} \overline{\exists} t''_j \in t_0 + [u, \infty). \\ \overline{\exists} t_{first,j} \in t_0 + [l, u).\overline{\exists} t_{last,j} \in t_0 + [l, u). \\ Q_1 t_1.Q_2 t_2.\ldots Q_{j-1} t_{j-1}. \end{matrix} \begin{Bmatrix} \forall t.t \ge t_{first,j} \wedge \\ S(t_{last,j}, t''_j) \wedge \\ \gamma_2 \end{Bmatrix}$$
- $C_{2,3}$ covers the possibility where there are points occurring within interval $(-\infty, l)$ but there is no point within $[u, \infty)$ from t_0. Hence,
$$C_{2,3} = \begin{matrix} \overline{\exists} t'_j \in t_0 + (-\infty, l). \\ \overline{\exists} t_{first,j} \in t_0 + [l, u).\overline{\exists} t_{last,j} \in t_0 + [l, u). \\ Q_1 t_1.Q_2 t_2.\ldots Q_{j-1} t_{j-1}. \end{matrix} \begin{Bmatrix} S(t'_j, t_{first,j}) \wedge \\ \forall t.t \le t_{last,j} \wedge \\ \gamma_2 \end{Bmatrix}$$
- $C_{2,4}$ covers the possibility where there is no point occurring within interval $(-\infty, l)$ and $[u, \infty)$ from t_0. Hence,
$C_{2,4} = \overline{\exists} t_{first,j} \in t_0 + [l, u).\overline{\exists} t_{last,j} \in I_j.Q_1 t_1.Q_2 t_2.\ldots Q_{j-1} t_{j-1}.(\forall t.t \ge t_{first,j} \wedge t \le t_{last,j} \wedge \gamma_2)$.

Hence, $C_1 \wedge C_2$ is the required formula. Note that irrespective of Q_j being a universal or existential quantifier, the new metric quantifiers that we add at the top level are only existential metric quantifiers. Hence, when we apply the above reduction for j steps, we will be able to get rid of all the $Q_1 \dots Q_j$ metric quantifiers (and hence the alternations within that block) and end up getting formulae where each time-constrained subformula contains a block of at most $4k$ existential metric quantifiers. □

6.3 Non-Adjacent GQMSO (NA-GQMSO)

Any AF-GQMSO formula φ is said to be **non-adjacent** if and only if for every subformula ψ of φ of the form $\overline{\exists} t_1 \in t + I_1 \dots \overline{\exists} t_j \in t + I_j \Phi(t, t_1, \dots, t_j)$, the set of intervals $\{I_1, \dots, I_j\}$ is non-adjacent. Notice that NA-GQMSO is a syntactic subclass of AF-GQMSO. For example, $\overline{\exists} t_1 \in t_0 + (2,3) \overline{\exists} t_2. \in t_0 + (3,4)[\exists t < t_0 \wedge \overline{\exists} t_3 \in t_0 + (4,5)]$ is not non-adjacent, as intervals $(2,3)$ and $(3,4)$ appear within the same metric quantifier block and are adjacent. However, $\overline{\exists} t_1 \in t_0 + (2,3) \overline{\exists} t_2. \in t_0 + (4,5)[\exists t < t_0 \wedge \overline{\exists} t_3 \in t_0 + (3,4)]$ is non-adjacent, as $\{(1,2),(4,5)\}$ and $(2,3)$ is non-punctual (and hence non-adjacent to itself). **The formula in Example 6.3 is also an NA-GQMSO formula.**

7 CLASSICAL LOGIC CHARACTERIZATION OF PnEMTL

In this section, we prove the following main theorem:

Theorem 7.1. *PnEMTL ≅ GQMSO. Moreover, Non-adjacent PnEMTL ≅ Non-adjacent GQMSO.*

The theorem follows from Lemmas 7.2 and 7.3 given below.

Lemma 7.2. ***PnEMTL ⊆ GQMSO.***

Proof. The key observation is that conditions of the form $\mathrm{Seg}(i, j, \rho, S) \in L(\mathsf{A})$ can be equivalently expressed as MSO[<] formulae $\psi_{\mathsf{A}}(i, j)$ using **Büchi Elgot Trakhtenbrot (BET)** Theorem [16, 30, 44]. Replacing the former with latter, we get an equivalent AF-GQMSO formula (which is a syntactic subset of GQMSO), as shown below. We apply induction on modal depth of the given formula φ. For modal depth 0, φ is a propositional formula and hence it is trivially an AF-GQMSO formula.

Let φ be a modal depth 1 formula of the form $\mathcal{F}^k_{I_1,\dots,I_k}(\mathsf{A}_1, \dots, \mathsf{A}_{k+1})(\Sigma)$. We can easily translate the above to equivalent GQMSO formula $\exists t_1 \in t + I_1 \dots \overline{\exists} t_j \in t + I_j\ \Phi(t, t_1, \dots, t_j)$, where $\Phi(t, t_1, \dots, t_j) = \exists t_{k+1}\ \psi_{A_1}(t_0, t_1) \wedge \cdots \wedge \psi_{A_k}(t_{k-1}, t_k) \wedge \psi_{A_{k+1}}(t_k, t_{k+1}) \wedge EP(t_{k+1})$. Note that the GQMSO formula directly encodes the semantics of $\mathcal{F}^k$ formula and hence their equivalence is clear by construction. The $\mathcal{P}^k$ modality is handled similarly. Also note that this reduction preserves the non-adjacency. Dealing with Boolean operators is trivial, as the AF-GQMSO is closed under Boolean operations.

For the induction step, we assume that the lemma holds for all the PnEMTL formulae of modal depth $< n$. Let $\varphi = \mathcal{F}^k_{I_1,\dots,I_k}(\mathsf{A}_1, \dots, \mathsf{A}_{k+1})(\Sigma \cup S)$ of modal depth n. Therefore, S is a set of PnEMTL formula with modal depth $< n$. We associate a unique new witness proposition with every subformulae in S and replace all the subformulae in S by their corresponding witness propositions getting a formula φ' of modal depth 1. As with the base case, we can construct an AF-GQMSO formula ψ' equivalent to φ'. By inductive hypothesis, every subformulae φ_i in S can be reduced to an equivalent AF-GQMSO formula ψ_i. We replace all the witnesses of φ_i by ψ_i getting an equivalent formulae ψ over Σ. Note that if formula φ_i in S are non- adjacent, then, by induction hypothesis, equivalent ψ_i are in NA-GQMSO formula. Similarly, if φ' is NA-PnEMTL formula, then ψ' is NA-GQMSO formula. Hence, if φ in non-adjacent, then equivalent formula ψ is non-adjacent, too. □

Lemma 7.3. ***GQMSO ⊆ PnEMTL.***

Proof. It suffices to show **AF-GQMSO ⊆ PnEMTL** (thanks to Theorem 6.4). The proof is done via induction on metric depth of the AF-GQMSO formulae.

(Base case) Let $\psi(t_0) = \overline{\exists} t_1 \in t_0 + I_1 \dots \overline{\exists} t_j \in t_0 + I_j.\varphi(t_0, t_1, \dots, t_j)$ be any AF-GQMSO formula of metric depth 1. Then, $\varphi(t_0, t_1, \dots, t_j)$ is an untimed MSO formula over $\Sigma_1 = \Sigma \cup \{t_0, \dots, t_j\}$. By Büchi Elgot Trakhtenbrot Theorem [16, 30, 44], we can construct a finite state automaton A_1 accepting same models as φ. Note that the alphabet of A_1 is 2^{Σ_1}. Every word α accepted by A_1 has exactly one position i_k where $t_k \in \alpha[j]$. Hence, with some abuse of notation, we can write $\alpha = \sigma \oplus (t_0 \partial i_0, t_1 \partial i_1, \dots, t_j \partial i_j)$ and $\sigma \in 2^{\Sigma}$. By the semantics of GQMSO, any pointed word $\rho, i \models \psi(t_0)$ iff $\exists i_1, i_2, \dots, i_j$ such that $\tau_i - \tau_{i_1} \in I_1 \wedge \dots \wedge \tau_i - \tau_{i_j} \in I_j$ and word $untime(\rho) \oplus (t_0 \partial i_0, t_1 \partial i_1, \dots, t_j \partial i_j)$ is accepted by A_1. We modify A_1 to give an $\mathcal{I}$ Interval word automaton A_2 as follows: If label of an edge is $S \subseteq \Sigma'$, then we relabel it with $S' \subseteq \Sigma \cup \{anch, I_1, \dots, I_j\}$, where $anch$ replaces t_0 and I_i replaces t_i in S. There is one-to-one correspondence between transitions of A_1 and A_2 where presence of interval I_i symbolically enforces the timing constraint. Hence, it is easy to see that $\rho, i \models \psi(t_0)$ iff $\rho, i \in \mathsf{Time}(L(A_2))$.

By the construction given in Section 5, for any NFA A over $\mathcal{I}$ interval words, we can construct a PnEMTL formulae $\varphi(A)$ such that for any pointed timed word ρ, i, we have $\rho, i \in Time(L(A))$ iff $\rho, i \models \varphi$. Hence, $\rho, i \models \psi(t_0)$ iff $\rho, i \in Time(L(A_2))$ iff $\rho, i \models \varphi(A_2)$. Moreover, if ψ is non-adjacent, then $\mathcal{I}$ is non-adjacent and thus φ is in NA-PnEMTL.

(Induction step) Assume that the lemma holds for all formulas of depth less than n. Let $\psi(t_0)$ be any time-constraint formula of AF-GQMSO having metric depth n. With every timed subformulae $\psi_i(t)$ of ψ, we associate a witness proposition b_i such that b_i holds iff ψ_i holds. Let W be the set of witnesses. We replace each subformula $\psi_i(t)$ of type MSO^T with its corresponding witness getting a formula $\psi'(t_0)$ of metric depth 1. As shown in the base case, we can construct a PnEMTL formula φ' equivalent to $\psi'(t_0)$ containing symbols from $\Sigma \cup W$. Note that all subformulae $\psi_i(t_0)$ of ψ are of metric depth less than n. Hence, by the induction hypothesis, we can construct a PnEMTL formula φ_i equivalent to $\psi_i(t_0)$. Hence, the witnesses for ψ_i are also that for φ_i. Replacing the witnesses b_i with its corresponding PnEMTL formulae φ_i, we get the required PnEMTL formulae φ. Also note that if ψ is non-adjacent, then all its subformulae ψ_i and formula ψ' are non-adjacent, too. This implies that formulae φ_i, φ' and, hence φ are NA-PnEMTL formulae. We give a small toy example as follows: In this example, we write a regular expression, in place of NFA wherever required, for the sake of succinctness and readability.

Example 7.4. Consider a GQMSO formulae $\psi(t) = \overline{\exists} t_1 \in t + (0,1) \overline{\exists} t_2 \in t + (-1, 0) \psi_{even,b}(t, t_1) \wedge \psi_{odd,a}(t, t_2)$, where $\psi_{even,b}(x, y)(\psi_{odd,a}(x, y))$ is an MSO[<] formula that is true iff the number of b's (a's, respectively) between x and y (including x and y) is even (odd, respectively). The regular expression of the behavior starting from the beginning would be of the form: $(\mathbf{a} + \mathbf{b})^* \cdot \{(\mathbf{a} + \mathbf{b}), \mathbf{x} \in (\mathbf{-1}, \mathbf{0})\} \cdot (\mathbf{b}^*.\mathbf{a}.\mathbf{b}^*.\mathbf{a}.\mathbf{b}^*)^* \cdot \mathbf{a} \cdot \mathbf{b}^*) \cdot \mathsf{anch} \cdot (\mathbf{a}^* \cdot \mathbf{b} \cdot \mathbf{a}^* \cdot \mathbf{b} \cdot \mathbf{a}^*) \cdot \{(\mathbf{a} + \mathbf{b}), \mathbf{x} \in (\mathbf{0}, \mathbf{1}))\} \cdot (\mathbf{a_b})^*$. By PnEMTL semantics, $\varphi = \mathcal{F}^1_{(0,1)}[(a^*.b.a^*.b.a^*), (a + b)^+](\{a, b\}) \wedge \mathcal{P}^1_{(0,1)}[(b^*.a.b^*.a.b^*)^*.a.b^*), (a + b)^+](\{a, b\})$ when asserted on a point t will accept the same set of behaviors. □

8 SATISFIABILITY CHECKING FOR NON-ADJACENT PNEMTL

The main result of the section is as follows:

THEOREM 8.1. *Satisfiability Checking for non-adjacent PnEMTL and non-adjacent 1-TPTL are decidable with EXPSPACE complete complexity. Satisfiability checking for NA-GQMSO is decidable.*

The proof is via a satisfiability-preserving reduction to logic $\text{EMITL}_{0,\infty}$ resulting in a formula whose size is at most exponential in the size of the input non-adjacent PnEMTL formula. Satisfiability checking for $\text{EMITL}_{0,\infty}$ is PSPACE complete [27]. This, along with our construction, implies an EXPSPACE decision procedure for satisfiability checking of non-adjacent PnEMTL. The

EXPSPACE lower bound follows from the EXPSPACE hardness of the sublogic MITL. The same complexity also applies to non-adjacent 1-TPTL, using the reduction in the Section 5. This also implies decidability for satisfiability checking of NA-GQMSO formulae, as they can be reduced to equivalent non-adjacent PnEMTL formulae (see Lemma 7.3) for which satisfiability could be checked using the following algorithm (reduction to equisatisfiable $\mathrm{EMITL}_{0,\infty}$ formulae). But the reduction in Lemma 7.3 incurs non-elementary blow-up. Hence, this results in a non-elementary decision procedure. This is to be expected, as lower-bound complexity for satisfiability checking for sublogic FO[<] is non-elementary.

We now describe the technicalities associated with our reduction. We use the technique of equisatisfiability modulo oversampling [31, 35]. Let Σ and OVS be disjoint set of propositions. Given any timed word ρ over Σ, we say that a word ρ' over $\Sigma \cup \mathsf{OVS}$ is an oversampling of ρ if $|\rho| \leq |\rho'|$ and when we delete the symbols in OVS from ρ', we get back ρ. Intuitively, OVS contains propositions that are used to label oversampling points only. Informally, a formulae α is equisatisfiable modulo oversampling to formulae β if and only if for every timed word ρ accepted by β there exists an oversampling of ρ accepted by α and, for every timed word ρ' accepted by α its projection is accepted by β. Note that when $|\rho'| > |\rho|$, ρ' will have some time points where no proposition from Σ is true. These new points are called oversampling points. Moreover, we say that any point $i' \in dom(\rho')$ is an old point of ρ' corresponding to i iff i' is the ith point of ρ' when we remove all the oversampling points. For the rest of this section, let ϕ be a non-adjacent PnEMTL formula over Σ. We break down the construction of an $\mathrm{EMITL}_{0,\infty}$ formula ψ as follows:

(1) Add oversampling points at every integer timestamp using φ_{ovs} below.
(2) Flatten the PnEMTL modalities to get rid of nested automata modalities, obtaining an equisatisfiable formula ϕ_{flat}.
(3) With the help of oversampling points, assert the properties expressed by PnEMTL subformulae ϕ_f of ϕ_{flat} using only $\mathrm{EMITL}_{0,\infty} + \mathsf{F}_{np}$ (F_I where I is restricted to be non-punctual) modalities getting formula ψ_f. This is done recursively as follows: Using the oversampling points:
  (a) For every $k > 1$ arity PnEMTL formula, construct an equivalent formula (for oversampled models) ψ_f^{k-1} with arity at most $k-1$.
  (b) For $k = 1$ arity formula construct an equivalent $\mathrm{EMITL}_0, \infty + \mathsf{F}_{np}$ modality.
(4) Finally, in ψ_f, only the F operators are timed with intervals of the form $\langle l, u\rangle$ where $0 < l < u < \infty$. We can reduce these time intervals into purely lower bound ($\langle l, \infty)$) or upper bound ($\langle 0, u\rangle$) constraints using these oversampling points, by reduction similar to that appearing in Reference [35], Chapter 5, Lemma 5.5.2, pp. 90–91, getting formula of size $O(\mathsf{cmax} \times |\psi_f|)$.

Let $\mathsf{Last} = \mathcal{G}\bot$ and $\mathsf{LastTS} = \mathcal{G}\bot \lor (\bot \mathsf{U}_{(0,\infty)} \top)$. Last is true only at the last point of any timed word. Similarly, LastTS, is true at a point i if there is no next point $i+1$ with the same timestamp τ_i. Let max be the maximum constant used in the intervals appearing in ϕ. Let $\mathsf{cmax} = max + 1$.

8.1 Behavior of Oversampling Points

We oversample timed words over Σ by adding new points where only propositions from Int holds, where $\mathsf{Int} \cap \Sigma = \emptyset$. Given a timed word ρ over Σ, consider an extension of ρ called ρ', by extending the alphabet Σ of ρ to $\Sigma' = \Sigma \cup \mathsf{Int}$. Compared to ρ, ρ' has extra points called *oversampling* points, where $\neg \bigvee \Sigma$ (and $\bigvee \mathsf{Int}$) hold. These extra points are added at all integer timestamps in such a way that, if ρ already has points with integer timestamps, then the oversampled point with the same timestamp appears last among all points with the same timestamp in ρ'. We will make use of these oversampling points to reduce the PnEMTL modalities into $\mathrm{EMITL}_{0,\infty}$. These oversampling points are labelled with a modulo counter $\mathsf{Int} = \{\mathsf{int}_0, \mathsf{int}_1, \ldots, \mathsf{int}_{\mathsf{cmax}-1}\}$. The counter is initialized to be 0 at the first oversampled point with timestamp 0 and is incremented, modulo cmax, after exactly one time unit till the last point of ρ. Let $i \oplus j = (i+j)\%\mathsf{cmax}$. The oversampled behaviors are expressed using the formula φ_{ovs}: $\{\neg \mathsf{F}_{(0,1)} \bigvee \mathsf{Int} \wedge \mathsf{F}_{[0,1)}\mathsf{int}_0\} \wedge \{\bigwedge_{i=0}^{\mathsf{cmax}-1} \mathcal{G}^w\{(\mathsf{int}_i \wedge \mathsf{F}(\bigvee \Sigma)) \rightarrow (\neg \mathsf{F}_{(0,1)}(\bigvee \mathsf{Int}) \wedge \mathsf{F}_{(0,1]}(\mathsf{int}_{i\oplus 1} \wedge (\neg \bigvee \Sigma) \wedge \mathsf{LastTS}))\}$. to an extension ρ' given by $\mathsf{ext}(\rho) = \rho'$ iff **(i)** ρ can be obtained from ρ' by deleting oversampling points and **(ii)** $\rho' \models \varphi_{\mathsf{ovs}}$. Map ext is well defined as for any ρ, $\rho' = \mathsf{ext}(\rho)$ if and only if ρ' can be constructed from ρ by appending oversampling points at integer timestamps and labelling kth such oversampling point (appearing at time $k-1$) with $\mathsf{int}_{k\%\mathsf{cmax}}$.

8.2 Flattening

Next, we flatten ϕ to eliminate the nested $\mathcal{F}^k_{\mathsf{I}_1,\ldots,\mathsf{I}_k}$ and $\mathcal{P}^k_{\mathsf{I}_1,\ldots,\mathsf{I}_k}$ modalities while preserving satisfiability. Flattening is well studied [27, 31, 35, 40]. The idea is to associate a fresh witness variable b_i to each subformula ϕ_i that needs to be flattened. This is achieved using the *temporal definition*, $T_i = \mathcal{G}^w((\bigvee \Sigma \wedge \phi_i) \leftrightarrow b_i)$, and replacing ϕ_i with b_i in ϕ, $\phi_i'' = \phi[b_i/\phi_i]$, where $\mathcal{G}^w$ is the weaker form of $\mathcal{G}$ asserting formula (within its scope) at the current point and all the strict future points. Then, $\phi_i' = \phi_i'' \wedge T_i \wedge \bigvee \Sigma$ is equisatisfiable to ϕ. Repeating this across all subformulae of ϕ, we obtain $\phi_{flat} = \phi_t \wedge T$ over the alphabet $\Sigma' = \Sigma \cup W$, where W is the set of the witness variables, $T = \bigwedge_i T_i$, ϕ_t is a propositional logic formula over W. Each T_i is of the form $\mathcal{G}^w(b_i \leftrightarrow (\phi_f \wedge \bigvee \Sigma))$, where $\phi_f = \mathcal{F}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}(\mathsf{A}_1,\ldots,\mathsf{A}_{n+1})(S)$ (or uses $\mathcal{P}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}$) and $S \subseteq \Sigma'$. For example, consider the formula $\phi = \mathcal{F}^2_{(0,1)(2,3)}(\mathcal{A}_1,\mathcal{A}_2,\mathcal{A}_3)(\{\phi_1,\phi_2\})$, where $\phi_1 = \mathcal{P}^2_{(0,2)(3,4)}(A_4,A_5,A_6)(\Sigma)$, $\phi_2 = \mathcal{P}^2_{(1,2)(4,5)}(A_7,A_8,A_9)(\Sigma)$. Replacing the ϕ_1,ϕ_2 modality with witness propositions b_1, b_2, respectively, we get $\phi_t = \mathcal{F}^2_{(0,1)(2,3)}(A_1,A_2,A_3)(\{b_1,b_2\}) \wedge T$, where $T = \mathcal{G}^w(b_1 \leftrightarrow (\bigvee \Sigma \wedge \phi_1)) \wedge \mathcal{G}^w(b_2 \leftrightarrow (\bigvee \Sigma \wedge \phi_2))$, A_1, A_2, A_3 are automata constructed from $\mathcal{A}_1,\mathcal{A}_2,\mathcal{A}_3$, respectively, by replacing ϕ_1 by b_1 and ϕ_2 by b_2 in the labels of their transitions. Hence, $\phi_{flat} = \phi_t \wedge T$ is obtained by flattening the $\mathcal{F}^k_{\mathsf{I}_1,\ldots,\mathsf{I}_k}, \mathcal{P}^k_{\mathsf{I}_1,\ldots,\mathsf{I}_k}$ modalities.

8.3 Constructing Equisatisfiable EMITL$_{0,\infty}$ Formula

In this step, for every PnEMTL formula ϕ_f appearing in each $T_i = \mathcal{G}^w(b_i \leftrightarrow (\phi_f \wedge \bigvee \Sigma))$, we will obtain an equisatisfiable EMITL$_{0,\infty}$ formula ψ_f. We use oversampling to construct the formula ψ_f such that for any timed word ρ over Σ, $i \in dom(\rho)$, there is an extension $\rho' = \mathsf{ext}(\rho)$ over an extended alphabet Σ', and a point $i' \in dom(\rho')$ that is an old point corresponding to i such that $\rho', i' \models \psi_f$ iff $\rho, i \models \phi_f$.

Consider $\phi_f = \mathcal{F}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}(\mathsf{A}_1,\ldots,\mathsf{A}_{n+1})(S)$ where $S \subseteq \Sigma'$. Without loss of generality, we assume:

- **[Assumption 1]:** $\inf(\mathsf{I}_1) \le \inf(\mathsf{I}_2) \le \cdots \le \inf(\mathsf{I}_n)$ and $\sup(\mathsf{I}_1) \le \cdots \le \sup(\mathsf{I}_n)$. This is w.l.o.g., since the check for A_{j+1} cannot start before the check of A_j in case of $\mathcal{F}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}$ modality (and vice versa for $\mathcal{P}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}$ modality) for any $1 \le j \le n$.
- **[Assumption 2]:** Intervals $\mathsf{I}_1, \ldots \mathsf{I}_{n-1}$ are bounded intervals. Interval I_n may or may not be bounded. This is also w.l.o.g.[11]

Let $\rho = (\sigma_1,\tau_1)\ldots(\sigma_n,\tau_n) \in T\Sigma^*$, $i \in dom(\rho)$. Let $\rho' = \mathsf{ext}(\rho)$ be defined by $(\sigma_1',\tau_1')\ldots(\sigma_m',\tau_m')$ with $m \ge n$, and each τ_x' is a either a new integer timestamp not among $\{\tau_1,\ldots,\tau_n\}$ or is some τ_y where x is an old action point corresponding to y. Let i' be an old point in ρ' corresponding to i. Let $i_0' = i'$ and $i_{n+1}' = |\rho'|$. As mentioned above, we make use of these extra action points in ρ' to assert specification same as ϕ_f without using EMITL$_{0,\infty}$ modalities (in case ϕ_f is arity 1 formula)

[11] Unbounded intervals can be eliminated using $\mathcal{F}^k_{\mathsf{I}_1,\mathsf{I}_2,\ldots,\mathsf{I}_{k-2},[l_1,\infty)[l_2,\infty)}(\mathsf{A}_1,\ldots,\mathsf{A}_{k+1}) \equiv \mathcal{F}^k_{\mathsf{I}_1,\mathsf{I}_2,\ldots,\mathsf{I}_{k-2},[l_1,\mathsf{cmax})[l_2,\infty)}(\mathsf{A}_1,\ldots,\mathsf{A}_{k+1}) \vee \mathcal{F}^{k-1}_{\mathsf{I}_1,\mathsf{I}_2,\ldots,\mathsf{I}_{k-2},[l_2,\infty)}(\mathsf{A}_1,\ldots,\mathsf{A}_{k-1},\mathsf{A}_k\cdot\mathsf{A}_{k+1})$.

or using PnEMTL modality with strictly smaller arity. We first construct a formula ϕ'_f in PnEMTL such that $\rho, i \models \phi_f$ iff $\rho', i' \models \phi'_f$. Note that the satisfaction of ϕ_f is sensitive to these extra action points of ρ'. Hence, $\rho, i \models \phi_f$ does not guarantee $\rho', i' \models \phi_f$ unless ϕ_f can be made to ignore oversampling points while checking for satisfaction. We do this as follows: For any $1 \leq j \leq n+1$, let A'_j be the automata built from A_j by adding self loop on $\neg \bigvee \Sigma$ (oversampling points) and $S' = S \cup \{\neg \bigvee \Sigma\}$. This self loop makes sure that A'_j ignores (or skips) all the oversampling points while checking for A_j. Hence, A'_j allows arbitrary interleaving of oversampling points while checking for A_j. We call such an NFA as **NFA relativized w.r.t.** Σ. Thus, we have the following proposition:

Proposition 8.2 (Relativization of Automata Modalities). *For any $g, h \in dom(\rho)$ with g', h' being old action points of ρ' corresponding to g, h, respectively,* $\mathrm{Seg}^s(\rho, g, h, S) \in L(\mathsf{A}_i)$ *iff* $\mathrm{Seg}^s(\rho', g', h', S \cup \{\neg \bigvee \Sigma\}) \in L(\mathsf{A}'_i)$ *for $s \in \{+,-\}$. Hence, $\rho, i \models \phi_f$ iff $\rho', i' \models \phi'_f$ where i' is an old action point of ρ' corresponding to i and $\phi'_f = \mathcal{F}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}(\mathsf{A}'_1, \ldots, \mathsf{A}'_{n+1})(S')$.*

From this point, we will work on eliminating PnEMTL modality from ϕ'_f rather than ϕ_f, as they are both equisatisfiable (if the former is restricted to be evaluated on models satisfying φ_{ovs}, i.e., oversampled models).

We present the reduction by applying induction on arity of the formula. That is, given a PnEMTL formula of arity k, we construct a formula of arity at most $k-1$ such that, for all timed words $\rho' \models \varphi_{ovs}$, for any old action point i' of ρ', $\rho', i' \models \phi'_f$ iff $\rho', i' \models \phi^{k-1}_f$ (Recursion Step). In other words, $\phi'_f \wedge \varphi_{ovs}$ is equivalent to $\phi^{k-1}_f \wedge \varphi_{ovs}$. Similarly, if ϕ'_f has arity 1, then we reduce it to an $\mathrm{EMITL}_{0,\infty}$ formula, ψ_f, such that $\phi'_f \wedge \varphi_{ovs}$ is equivalent to $\psi_f \wedge \varphi_{ovs}$ (Base Step). We start with the latter (Base step). That is, we assume that $\phi'_f = \mathcal{F}_{I_1}(\mathsf{A}'_1, \mathsf{A}'_2)(S')$, we construct a formula ψ_f such that ψ_f only contains $\mathrm{EMITL}_{0,\infty}$ modalities. Before starting with the reduction, we state some useful notations and lemma. For the sake of readability, from this point onward, we do not explicitly mention set of formulae over which the automata modalities are being evaluated unless it is not clear from the context. For example, $\phi'_f = \mathcal{F}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}(\mathsf{A}'_1, \ldots, \mathsf{A}'_{n+1})(S \cup \neg(\bigvee \Sigma\})$ will be simply written as $\mathcal{F}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}(\mathsf{A}'_1, \ldots, \mathsf{A}'_{n+1})$.

8.3.1 Notations. Let $\mathsf{A} = (Q, q_0, 2^\Sigma, \delta, F)$ be any NFA. For any $q_1 \in Q, Q_2 \subseteq Q$, $\mathsf{A}[q_1, Q_2]$ denotes NFA $(Q, q_1, 2^\Sigma, \delta, Q_2)$. For the sake of readability, we abuse this notation by denoting $\mathsf{A}[q_1, \{q_2\}]$ as $\mathsf{A}[q_1, q_2]$ for any $q_2 \in Q$. Rev(A) denotes the NFA accepting the language that is reverse of A. Similarly, $\mathsf{A} \cdot X$ for any set of propositions X denotes an NFA $(Q \cup f, q_0, 2^{\Sigma \cup X}, \delta', \{f\})$ where $\delta' = \delta \cup \{(q, X, f) | q \in F\}$. In other words, $\mathsf{A} \cdot X$ is an NFA that accepts all the words $w \cdot X$ where w is accepted by A. Similarly, for any two automata A and A′, $\mathsf{A} \cdot \mathsf{A}'$ denotes NFA constructed by concatenating A with A′. Let $a \notin \Sigma$. We define $\mathsf{A}^a = (Q, q_0, 2^{\Sigma \cup \{a\}}, \delta^a, F)$ where $\delta^a = \{(q, W, q'), (q, W \cup \{a\}, q') | (q, W, q') \in \delta\}$. Hence, for any $g, h \in dom(\rho)$, $\mathrm{Seg}^{+/-}(\rho, g, h, \Sigma) \in L(\mathsf{A}) \iff \mathrm{Seg}^{+/-}(\rho, g, h, \Sigma \cup \{a\}) \in L(\mathsf{A}^a)$. Hence, A_a behaves exactly like A irrespective of the occurrence or absence of a at any point. Similarly, we define $\mathsf{A}^{last,a} = (Q \cup F^a, q_0, 2^{\Sigma \cup \{a\}}, \delta^{last,a}, F^a)$ where $F^a = \{(q, 1) | q \in F\}$, $\delta^{last,a} = \delta^a \cup \{(q, W, (q', 1)) | q' \in F \wedge a \in W \wedge (q, W \setminus \{a\}, q') \in \delta^a\}$. In other words, $L(\mathsf{A}^{last,a}) = L(\mathsf{A}^a) \cap (\bigcup_{W \subseteq \Sigma} L((2^\Sigma)^* \cdot (W \cup \{a\})))$. Hence, the $\mathsf{A}^{last,a}$ accepts exactly those words w accepted by A^a whose last letter contains proposition a. Note that all these operations result in an NFA that is linear in the size of the input NFA(s).

8.3.2 Lemma for Factoring Regular Languages. As mentioned above, we fix $\phi_f = \mathcal{F}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}(\mathsf{A}_1, \ldots, \mathsf{A}_{n+1})$ and $\phi'_f = \mathcal{F}^n_{\mathsf{I}_1,\ldots,\mathsf{I}_n}(\mathsf{A}'_1, \ldots, \mathsf{A}'_{n+1})$, where $\mathsf{A}'_1, \ldots, \mathsf{A}'_{n+1}$ are NFA relativized w.r.t. Σ. The case of $\mathcal{P}^k_{\mathsf{I}_1,\ldots,\mathsf{I}_k}$ modality can be handled symmetrically. We fix $\rho = (\sigma_1, \tau_1) \ldots (\sigma_m, \tau_m)$

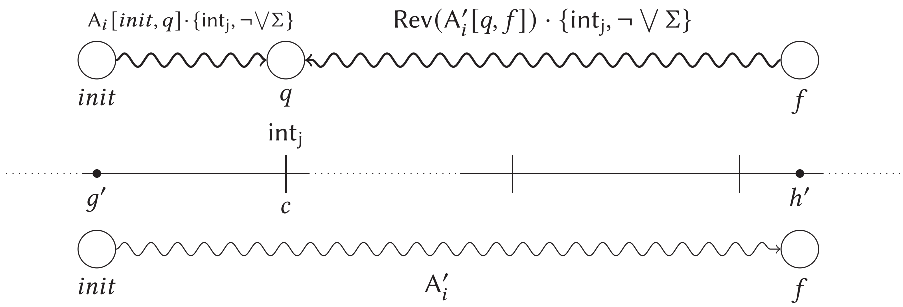

Fig. 8. Figure for Lemma 8.3. For any word ρ' satisfying φ_{ovs}, checking whether pattern of satisfaction of subformulae in S' between points g' and h' is accepted by A'_i can be reduced to asserting untimed EMITL formulae at g' and h'. The behavior from g' to c is given by $\mathcal{F}(A_i[init, q].\{int_j, \neg \bigvee \Sigma\})$ for some $q \in Q_i$ and the corresponding behavior from c to h' is given by $\mathcal{P}(Rev(A'_k[q, f]) \cdot \{int_j, \neg \bigvee \Sigma\})$ for some final state $f \in F_i$. Disjuncting over all possible (but finitely many) $j \in \{0, cmax - 1\}$, $q \in Q_i$ and $f \in F_i$, we get the required formulae.

and $\rho' = (\sigma'_1, \tau'_1) \dots (\sigma'_{m'}, \tau'_{m'})$ =ext(ρ). Let i be any arbitrary point of ρ and i' be an old action point of ρ' corresponding to i. $i'_0 = i'$. We first present a lemma that reduces the check for condition of the form $Seg^+(\rho', g', h', S') \in L(A'_i)$ by asserting some $EMITL_{0,\infty}$ formulae at g' and h' for any $i \in \{1, \dots, n+1\}$. For any $g', h' \in dom(\rho)$, let us call segments of the form $Seg^+(\rho', g', h', S')$ or $Seg^-(\rho', g', h', S')$ as Segments of ρ' over S'. Let $i \in \{1, \dots, n+1\}$ be any integer.
Remark: As mentioned above, when A'_i is evaluated over segments of ρ' over S', it skips all the oversampling points. But note that the same A'_i when evaluated over segments of $S' \cup \{int_j\}$ for some $0 \le j <$ cmax skips all oversampling points except those labelled with int_j. This is because the transitions of A'_i are labelled using subformulae in S' that do not contain any symbols from Int. Hence, A'_i has no transition on symbol int_j. Thus, $Seg^+(\rho', g', h', S') \in L(A'_i \cdot \{int_j, \neg \bigvee \Sigma\})$ iff $Seg^+(\rho', g', h' - 1, S') \in L(A'_i)$, none of the points between g' to $h' - 1$ are labelled with int_j and point h' is labelled with proposition int_j. Hence, h' is the first point after g' where int_j holds.
Let $A'_i = (Q_i, init_i, S', \delta, F_i)$.

Lemma 8.3 (Factoring Check for Regular Language). *Let g', h' be any two points of ρ' such that $g' < h'$, $\tau'_{h'} - \tau'_{g'} \le$ cmax and $\lceil \tau'_{g'} \rceil \ne \lceil \tau'_{h'} \rceil$. Then,* $Seg^+(\rho', g', h', S') \in L(A'_i)$ *iff* $\bigvee_{j=0}^{cmax-1} \bigvee_{q \in Q_i} \bigvee_{f \in F_i} [\rho', g' \models \psi^+(i, init_i, q, j) \wedge \rho', h' \models \psi^-(i, q, f, j)$, *where* $\psi^+(i, init_i, q, j) = \mathcal{F}(A_i[init_i, q].\{int_j, \neg \bigvee \Sigma\})(S' \cup \{int_j\})$ *and* $\psi^-(i, q, f, j) = \mathcal{P}(Rev(A'_i[q, f]).\{int_j, \neg \bigvee \Sigma\})(S' \cup \{int_j\})]$.

Proof. **Intuition:** We encourage readers to look at Figure 8. As mentioned above, the main purpose of this lemma is to reduce the checking of condition $Seg^+(\rho', g', h', S') \in L(A'_i)$ by asserting some $EMITL_{0,\infty}$ formulae at g' and h'. As ρ' satisfies φ_{ovs} and $\tau'_{h'} - \tau'_{g'} \le$ cmax, all the oversampling integer points between g' and h' are labelled with unique counters, as the counters increment at every oversampling point modulo cmax. Hence, if the oversampling integer time point immediately after g' is labelled int_j, then no other point between g' and h' is labelled int_j. Moreover, the oversampling integer time point immediately after g' (say, c) is labelled with a proposition int_j iff $\lceil \tau'_{g'} \rceil \%$cmax $= j$. For checking $Seg^+(\rho', g', h', S') \in L(A'_i)$, we make use of this oversampling point c to split the run(s) as follows:

(1) **Checking the First Part:** Concretely, checking for $Seg^+(\rho', g', h', S') \in L(A'_i)$, we start at g' in ρ', from the initial state $init_i$ of A_i, and move to the state (say, q) that is reached at

the closest oversampling point c. Note that we use only A_i (without any $\neg \bigvee \Sigma$ self loops) to disallow occurrence of any oversampling point except at the last point. This ensures that we end our run after reading the closest oversampling point c.

(2) **Checking the Latter Part**: Reaching q from $init_i$, we have read a partial behavior between g' and c; this must be extended to check full behavior by starting from state q, continuing from point c, with transition rules of A'_i and assert that we end at an accepting state after reading the point h'. Note that we use A'_i instead of A_i (used in the first part) to ignore the oversampling points that could be encountered while checking the latter part, i.e., from c to h'). Hence, starting from g' with initial state of A'_i, we reach at the accepting state of A'_i after reading point h' iff we end at some state q after the end of checking the first part while simulating A_i, after which on simulating A'_i and continuing from state q, we reach some accepting state of A'_i on reading till h' and hence ending the check for the second part.

Note that check for the first part ending at some state q can be characterized by $\rho', g' \models \mathcal{F}(\mathsf{A}_i[init, q].\{\mathsf{int}_j, \neg \bigvee \Sigma\})$. For reducing the check of latter part with a formula asserting at h', we start the check for automata in reverse. That is, we assert that: Starting from some final state f from h', if we simulate the A'_i in reverse direction till point c, then we should be able to reach q. Note that the end point of the segment in $\mathrm{EMITL}_{0,\infty}$ formula is within an existential quantifier. Then, how do we make sure that we end our check at c? This can be done by asserting that the check ends at the nearest point before h' where int_j holds first. As c is the only point between g' and h' where int_j holds, we are sure to end at c. Hence, checking for latter part is equivalent to check $\rho', h' \models \mathcal{P}(\mathsf{Rev}(\mathsf{A}'_i[q, f]).\{\mathsf{int}_j, \neg \bigvee \Sigma\})(S' \cup \{\mathsf{int}_j\})$. Before starting the proof, we give a very simple example that gives some intuition about the construction.

Example 8.4. Consider the formula $\phi = \mathcal{F}^2_{(1,2),(3,4)}(Even_a, b^*, \Sigma^*)$, where $Even_a$ is an automaton accepting strings containing even number of $a's$. ρ, i satisfies ϕ if and only if there exist points i_1 (within $(1, 2)$ of i) and i_2 (within $(3, 4)$ of i) such that there is an even number of $a's$ between i and i_1 and only $b's$ occur between i_1 and i_2. Observe the following Figure 9. Consider $\rho' = \mathsf{ext}(\rho)$. Let i', i'_1, i'_2 be the points of ρ' corresponding to old action points i, i_1, i_2 of ρ, respectively. Now, we can break the check between i' and i'_1 at the smallest integer oversampling point occurring after i' labelled int_j. The number of $a's$ between i' and i'_1 (and hence number of $a's$ between i and i_1) is even iff the number of $a's$ between i' and next occurring int_j and the number of $a's$ between int_j and i'_1 are either both odd or both even. Similarly, all points between i_1 and i_2 are labelled b iff at all old action points between i'_1 and nearest point x labelled $\mathsf{int}_{j'}$ (where $j' \in \{0, \ldots, \mathsf{cmax} - 1\}$), and continuing from x to i_2 only b or $\mathsf{int}_{j''}$ occurs where $j'' \neq j'$. Hence, formula ϕ is satisfiable iff the following formula ψ is satisfiable:

$$\psi = \varphi_{ovs} \wedge \bigvee_{0 \le j, j' \le \mathsf{cmax}-1, j'' \neq j'} [\mathsf{F}_{(0,1]}(\mathsf{int}_j) \rightarrow [\{\mathsf{F}_{(3,4)}(\mathcal{P}((b + \mathsf{int}_{j''})^*.\mathsf{int}j'))\} \wedge \{\{\mathcal{F}(Even_a.\mathsf{int}_j) \wedge \mathsf{F}_{(1,2)}(\mathcal{P}(Even_a.\mathsf{int}_j) \wedge \mathcal{F}(b^*.\mathsf{int}_{j \oplus 2})\} \vee \{\mathcal{F}(Odd_a.\mathsf{int}_j) \wedge \mathsf{F}_{(1,2)}(\mathcal{P}(Odd_a.\mathsf{int}_j) \wedge \mathcal{F}(b^*.\mathsf{int}_{j'})))\}\}]].$$

Formal Proof: We argue for correctness as follows:

(1) Let c' be any point between g' and h'. Then, any accepting run from point g' to h' ending at an accepting state $f \in F$ will pass through point c' such that after reading the point c' the run ends up at some state q. Thus, the behavior from g' to c' is given by all the runs starting from $init$ and ending at state q (hence in $\mathsf{A}'_i[init, q]$). Similarly, the remaining part of the run from $c' + 1$ to h' is characterized by those continuing from q to f (hence, in $\mathsf{A}'_i(q, f)$). Disjuncting over all possible values of $q \in Q$ and $f \in F$, we get all the possible accepting

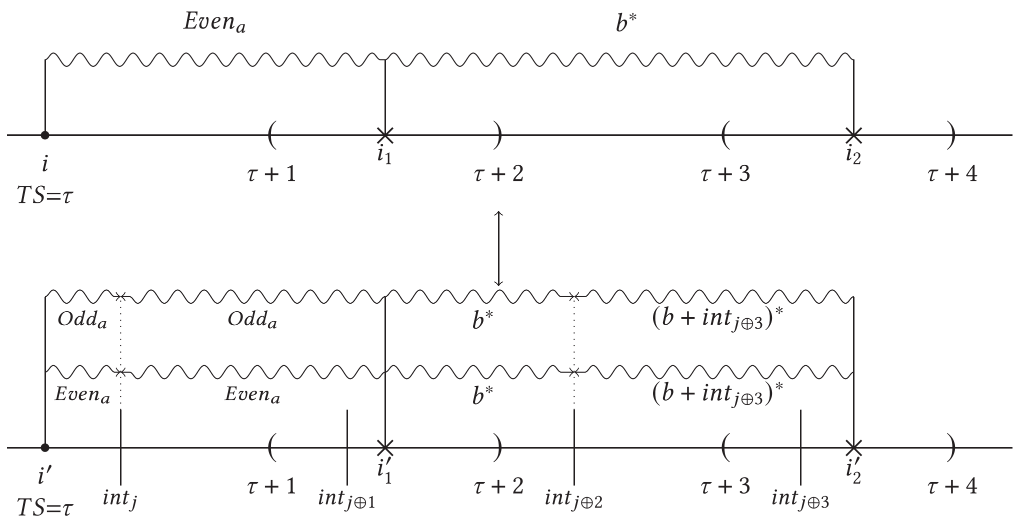

Fig. 9. Figure depicting the construction of equisatisfiable EMITL formula from non-adjacent PnEMTL formula. TS stands for timestamp. Hence, τ is a timestamp of point i_0. At the top, we have timed word ρ and the bottom part of the figure denotes $\rho' = \text{ext}(\rho)$.

runs. Hence, $\forall c'.g' < c' < h'$, we have

$$\mathrm{Seg}^+(\rho', g', h', S') \in L(\mathsf{A}'_i) \iff \bigvee_{q \in Q_i} \bigvee_{f \in F_i} \begin{array}{l} \mathrm{Seg}^+(\rho', g', c', S') \in L(\mathsf{A}'_i[init, q]) \\ \wedge \mathrm{Seg}^+(\rho', c'+1, h', S') \in L(\mathsf{A}'_i[q, f]). \end{array} \tag{1}$$

(2) As $\lceil \tau'_{g'} \rceil \neq \lceil \tau'_{h'} \rceil$ and $g' < h'$, $\tau'_{h'} > \lceil \tau'_{g'} \rceil$. Moreover, as $\rho' \models \varphi_{\text{ovs}}$, there is an oversampling point c with timestamp $\tau'_c = \lceil \tau'_{g'} \rceil$ where int_j holds. Hence, by Equation (1) and as $g' \leq c \leq h'$, we have

$$\mathrm{Seg}^+(\rho', g', h', S') \in L(\mathsf{A}'_i) \iff \bigvee_{q \in Q_i} \bigvee_{f \in F_i} \begin{array}{l} \mathrm{Seg}^+(\rho', g', c, S' \cup) \in L(\mathsf{A}'_i[init, q]) \\ \wedge \mathrm{Seg}^+(\rho', c+1, h', S') \in L(\mathsf{A}'_i[q, f]). \end{array} \tag{2}$$

(3) As A'_i has a self loop over $\neg \bigvee \Sigma$, the states do not change on reading (or not reading) the oversampling point c. Hence, $\mathrm{Seg}^+(\rho', g', c, S') \in L(\mathsf{A}'_i[q, Q_f]) \iff \mathrm{Seg}^+(\rho', g', c-1, S') \in L(\mathsf{A}'_i[q, Q_f])$. This implies:

$$\mathrm{Seg}^+(\rho', g', h', S') \in L(\mathsf{A}'_i) \iff \bigvee_{q \in Q} \bigvee_{f \in F} \begin{array}{l} \mathrm{Seg}^+(\rho', g', c-1, S') \in L(\mathsf{A}'_i[init, q]) \\ \wedge \mathrm{Seg}^+(\rho', c+1, h', S') \in L(\mathsf{A}'_i[q, f]). \end{array} \tag{3}$$

(4) By definition of Seg^+ and Seg^-, $\mathrm{Seg}^+(\rho', c+1, h', S') \in L(\mathsf{A}'_i[q, f]) \iff \mathrm{Seg}^-(\rho', h', c+1, S') \in L(\mathrm{Rev}(\mathsf{A}'_i[f, q]))$. This, along with Equation (3), implies,

$$\mathrm{Seg}^+(\rho', g', h', S') \in L(\mathsf{A}'_i) \iff \bigvee_{q \in Q_i} \bigvee_{f \in F_i} \begin{array}{l} [\mathrm{Seg}^+(\rho', g', c-1, S') \in L(\mathsf{A}'_i[init_i, q]) \\ \wedge \mathrm{Seg}^-(\rho', h', c+1, S') \in L(\mathrm{Rev}(\mathsf{A}'_i[q, f])]. \end{array} \tag{4}$$

(5) As $\rho' \models \varphi_{\text{ovs}}$. If $j = \lceil \tau'_{g'} \rceil \% \text{cmax}$, then point c is labelled with int_j. Moreover, there is no oversampling point between g' and c. Hence,

$$\begin{aligned} \text{Seg}^+(\rho', g', c-1, S') \in L(\mathsf{A}'_i[init_i, q]) \iff \exists g' < c'. \\ \text{Seg}^+(\rho', g', c', S' \cup \{\text{int}_j\}) \in L\left(\mathsf{A}_i[init_i, q] \cdot \{\text{int}_j, \bigvee \Sigma\}\right). \end{aligned} \tag{5}$$

Note that we use A_i instead of A'_i, as A_i will make sure that in the initial part from g' to $c'-1$ there is no oversampling point, as it has no self loops on $\neg \bigvee \Sigma$. This will ensure that the c' point is the very first oversampling point after g'. Hence, there is only one choice for c', i.e., c. Moreover, concatenating $\{\text{int}_j, \bigvee \Sigma\}$ at the end makes sure that c is labelled as int_j.

(6) Similarly,

$$\begin{aligned} \text{Seg}^-(\rho', h', c+1, S') \in L(\mathsf{A}'_i[q, f]) \iff \exists c' < h'. \\ \text{Seg}^-(\rho', h', c', S' \cup \{\text{int}_j\}) \in L\left(\text{Rev}\left(\mathsf{A}'_i[q, f]\right) \cdot \left\{\text{int}_j, \bigvee \Sigma\right\}\right). \end{aligned} \tag{6}$$

As A'_i does not contain symbols from Int, c' is the nearest such point before h' where int_j holds. As $\tau'_{h'}$ - $\tau'_{g'} \leq$ cmax and the counters are incremented modulo cmax at integer timestamps by φ_{ovs}, if c is labelled as int_j, then there is no other point between g' and h' that will be labelled int_j. Hence, there is only one choice for c', i.e., c.

(7) By semantics of $\mathcal{F}$ and Equation (5), we have:

$$\text{Seg}^+(\rho', g', c-1, S') \in L(\mathsf{A}'_i[init_i, q]) \iff \rho', g' \models \mathcal{F}\left(\mathsf{A}_i[init_i, q] \cdot \left\{\text{int}_j, \bigvee \Sigma\right\}\right). \tag{7}$$

Similarly, by semantics of $\mathcal{P}$ and Equation (6), we have:

$$\text{Seg}^-(\rho', h', c+1, S') \in L(\mathsf{A}'_i[q, f]) \iff \rho', h' \models \mathcal{P}\left(\text{Rev}\left(\mathsf{A}_i[q, f]\right) \cdot \left\{\text{int}_j, \bigvee \Sigma\right\}\right). \tag{8}$$

(8) By Equations (4), (7), and (8) and disjuncting over all possible values of $q \in Q_i$ and $f \in F_i$, if $j = \lceil \tau'_{g'} \rceil \% \text{cmax}$, we have:

$$\text{Seg}^+(\rho', g', h', S') \in L(\mathsf{A}'_i) \iff \bigvee_{q \in Q} \bigvee_{f \in F} [\rho', g' \models \psi^+(i, init_i, q, j) \wedge \rho', h' \models \psi^-(i, q, f, j). \tag{9}$$

Finally, disjuncting over all possible values of $j \in \{0, \ldots, \text{cmax}-1\}$, we have the required result:

$$\text{Seg}^+(\rho', g', h', S') \in L(\mathsf{A}'_i) \iff \bigvee_{j=0}^{\text{cmax}-1} \bigvee_{q \in Q} \bigvee_{f \in F} [\rho', g' \models \psi^+(i, init_i, q, j) \wedge \rho', h' \models \psi^-(i, q, f, j)]. \tag{10}$$

□

We now start with the base case of the construction.

Lemma 8.5 (Base Lemma). *If $\phi'_f = \mathcal{F}^1_{\langle l,u \rangle}(\mathsf{A}'_1, \mathsf{A}'_2)(S')$(i.e., $n = 1$), then we can construct an EMITL$_{0,\infty}$ formula ψ_f such that ρ', i' satisfies ϕ'_f if and only if it satisfies ψ_f. Moreover, the total number of operators (temporal and Boolean), $N = O(\text{cmax} \times |\mathsf{A}_1| \times |\mathsf{A}_1| \times |\phi_f|)$.*

Proof. To reiterate the semantics of ϕ'_f:

$$\rho', i' \models \phi'_f \iff \exists i'_1 . \tau_{i'_1} - \tau_{i'} \in \langle l, u \rangle \wedge \text{Seg}^+(\rho', i'_0, i'_1, S') \in L(\mathsf{A}'_1) \wedge \text{Seg}^+(\rho', i'_1, m', S') \in L(\mathsf{A}'_2). \tag{11}$$

- **Case 1:** $l = 0$ or $u = \infty$.
  Intuition: This case is straightforward. As we have only PnEMTL modality with unit arity and the intervals are either of the form $\langle 0, u \rangle$ or $\langle l, \infty \rangle$, we can use an $\mathcal{F}_{\langle l,u \rangle}$ EMITL$_{0,\infty}$ formula to assert the check for A'_1, which has a nested untimed EMITL formulae to check A'_2.

Formal Construction and proof: In this case, we can trivially reduce ϕ'_f into an equivalent ψ_f that is already in $\text{EMITL}_{0,\infty}$ using nesting: $\psi_f = \mathcal{F}_{\langle l,u\rangle}({\mathsf{A}'_1}^{\text{last},\beta})$ where $\beta = \mathcal{F}(\mathsf{A}'_2 \cdot \text{Last})(S' \cup \{\text{Last}\})$. By semantics of $\mathcal{F}$ modality,

$$\rho', i' \models \psi_f \iff \exists i'_1.\tau_{i'_1} - \tau_{i'} \in \langle l,u\rangle \wedge \text{Seg}^+(\rho', i', i'_1, S' \cup \{\beta, \text{last}\}) \in L({\mathsf{A}'_1}^{\text{last},\beta}). \quad (12)$$

Moreover, by definition of $\mathsf{A}^{last,a}$,

$$\rho', i' \models \psi_f \iff \tau_{i'_1} - \tau_{i'} \in \langle l,u\rangle \wedge \text{Seg}^+(\rho', i', i'_1, S') \in L(\mathsf{A}'_1) \wedge \rho', i'_1 \models \beta. \quad (13)$$

Note that

$$\rho', i'_1 \models \beta \iff \text{Seg}^+(\rho', i'_1, m'-1, S') \in L(\mathsf{A}'_2) \text{ (by semantics)} \iff \text{Seg}^+(\rho', i'_1, m', S') \in L(\mathsf{A}'_2). \quad (14)$$

The equivalence in the right is due to the observation the last point is always an oversampling point, as $\rho' \models \varphi_{\text{ovs}}$, and A'_2 loops over oversampling points when evaluated on segments over S' (hence, the set of states reached at $m'-1$ and m are the same). By Equations (14) and (13), we get $\rho', i' \models \phi'_f \iff \rho', i' \models \psi_f$, where ψ_f is an $\text{EMITL}_{0,\infty}$ formula. Note that in this case $|\psi_f| = O(|\phi_f|)$.

- **Case 2:** $\langle l,u\rangle$ is a bounded interval where $l > 0$. Hence, $\text{cmax} \geq \tau'_{i'_1} - \tau'_{i'} \geq 1$. As $\tau'_{i'_1} - \tau'_{i'} \geq 1$ implies $\lceil \tau'_{i'_1} \rceil \neq \lceil \tau'_{i'} \rceil$, we can apply the Lemma 8.3. Let $\mathsf{A}_1 = (Q, init, 2^{S'}, \delta, F)$.

  Intuition: In this case, to check $\text{Seg}^+(\rho', i'_0, i'_1, S') \in L(\mathsf{A}'_1)$, we use Lemma 8.3, which gives us $\text{EMITL}_{0,\infty}$ $\mathcal{F}$ formulae (of the form ψ^+, as mentioned in Lemma 8.3) to be asserted at i' and $\mathcal{P}$ formulae of the form ψ^- to be asserted at i'_1. The former can be asserted directly, as i' is the present point. For asserting formulae at i'_1, we jump from i' to i'_1 using $\mathsf{F}_{\langle l,u\rangle}$ modality and assert the corresponding $\mathcal{P}$ modality. For checking A'_2, we assert formulae β, as constructed in the previous case at i'_1.

  Formal Construction and Proof:

$$\text{Seg}^+(\rho', i'_0, i'_1, S') \in L(\mathsf{A}'_1) \iff \bigvee_{j=0}^{\text{cmax}-1} \bigvee_{q\in Q} \bigvee_{f\in F} [\rho', g' \models \psi^+(1, init, q, j) \wedge \rho', h' \models \psi^-(1, q, f, j)] \quad (15)$$

Using Equations (15) and (14) in Equation (11), we get:

$$\rho', i' \models \phi'_f \iff$$

$$\exists i'_1.\tau_{i'_1} - \tau_{i'} \in \langle l,u\rangle \wedge \bigvee_{j=0}^{\text{cmax}-1} \bigvee_{q\in Q} \bigvee_{f\in F} [\rho', i'_0 \models \psi^+(1, init, q, j) \wedge \rho', i'_1 \models \psi^-(1, q, f, j)] \wedge \rho', i'_1 \models \beta, \quad (16)$$

where β is the same as one used in Equation (14). We can eliminate the quantifier guarded by the timing constraint $\exists i'_1.\tau_{i'_1} - \tau_{i'} \in \langle l,u\rangle$ using $\mathsf{F}_{\langle l,u\rangle}$ modality. Hence, by semantics of $\mathsf{F}_{\langle l,u\rangle}$ modality, if $\psi_f = \bigvee_{j=0}^{\text{cmax}-1} \bigvee_{q\in Q} \bigvee_{f\in F} \psi^+(1, init, q, j) \wedge \mathsf{F}_{I_1}(\psi^-(1, q, f, j) \wedge \beta]$, then by semantics of $\rho', i' \models \phi'_f \iff \rho', i' \models \psi_f$. Note that ψ_f contains only $\text{EMITL}_{0,\infty}$ and F_{np} modalities and hence is the required formulae. Also note that, as each $\psi^+(1, init, q, j)$ and $\psi^-(1, q, f, j)$ formulae are of the size of $O(\phi_f)$, we have $|\psi_f| = O(\text{cmax} \times |Q_1| \times |F_1| \times |\phi_f|)$. □

We now give the recursive reduction. We show that, given any non-adjacent PnEMTL formula of arity k, we can construct an equisatisfiable formulae non-adjacent PnEMTL formula of arity $k-1$ or less.

Lemma 8.6 (Recursive Reduction Lemma). *If $\phi'_f = \mathcal{F}^k_{I_1,\dots I_k}(\mathsf{A}'_1,\dots,\mathsf{A}'_{k+1})$, then we can construct an PnEMTL formula ψ^{k-1}_f with arity at most $k-1$ such that ρ', i' satisfies ϕ'_f if and only if it satisfies ψ^{k-1}_f. Moreover, the total number of operators $N = O(\mathsf{cmax} \times |\mathsf{A}_k| \times |\mathsf{A}_k| \times |\phi_f|)$.*

Proof. To rephrase the semantics of $\rho', i' \models \phi'_f$ (by pushing $\exists i'_{k-1} \le i'_k.(\tau'_{i'_k} - \tau'_{i'} \in \mathsf{I}_k$ inside):

$$\rho', i' \models \phi'_k \Longleftrightarrow \exists i' \le i'_1 \le \dots \le i'_{k-1}. \bigwedge_{g=1}^{k-1} (\tau'_{i'_g} - \tau'_{i'} \in \mathsf{I}_g \wedge \rho', i'_g \models \bigvee \Sigma \wedge \mathrm{Seg}^+(\rho', i'_{g-1}, i'_g, S') \in L(\mathsf{A}'_g))$$
$$\wedge\ \exists i'_{k-1} \le i'_k.(\tau'_{i'_k} - \tau'_{i'} \in \mathsf{I}_k \wedge \mathrm{Seg}^+(\rho', i'_{k-1}, i'_k, S') \in L(\mathsf{A}'_k) \wedge \mathrm{Seg}^+(\rho', i'_k, m', S') \in L(\mathsf{A}'_{k+1}). \tag{17}$$

Let $\mathsf{A}'_k = (Q, \mathsf{init}, 2^{S'}, \delta, F)$, $I_{k-1} = \langle l_{k-1}, u_{k-1}\rangle$ and $I_k = \langle l_k, u_k\rangle$.

- **Case 1:** I_{k-1} and I_k are non-overlapping. That is, $u_{k-1} < l_k$ (strict $<$ is implied by the fact that the set of intervals $\{\mathsf{I}_1,\dots,\mathsf{I}_\mathsf{k}\}$ is non-adjacent). Hence, $\tau'_{i'_{k-1}} - \tau'_{i'_k} \ge 1$ for any possible value of i'_{k-1} and i'_k.
  - **Case 1.1:** I_k is bounded, i.e., $u_k \ne \infty$. Then, $\tau'_{i'_{k-1}} - \tau'_{i'_k} \le \mathsf{cmax}$ holds.
    Intuition: Refer to Figure 10. Similar to case 2 of Lemma 8.5, we apply Lemma 8.3 to split the check for $\mathrm{Seg}^+(\rho', i'_{k-1}, i'_k, S') \in L(\mathsf{A}'_k)$ at the nearest oversampling point c after i'_{k-1}. The first part of the check (from i'_{k-1} to c) can be asserted using the kth tail automata of $k-1$-ary PnEMTL formula, where the first $k-2$ arguments are identical to that of ϕ'_f. The second part of the check (from c to i'_k) can be asserted in the reverse direction from i'_k by jumping to it from i' using F_{I_k} modality.
    Construction: By Lemma 8.3,

$$\mathrm{Seg}^+(\rho', i'_{k-1}, i'_k, S') \in L(\mathsf{A}'_k) \equiv \bigvee_{j=0}^{\mathsf{cmax}-1} \bigvee_{q \in Q} \bigvee_{f \in F} \rho', i'_{k-1} \models \exists c'.Seg^+(\rho', i'_{k-1}, c', S')$$
$$\in L(\mathsf{A}_k[init, q] \cdot \mathsf{int}_j) \wedge \rho', i'_k \models \psi^-(k, q, f, j). \tag{18}$$

For the sake of brevity, we make following abuses of notation: For any NFA A and any proposition $\mathsf{int}_j \in \mathsf{Int}$, $\mathsf{A} \cdot \{nt_j, \neg \bigvee \Sigma\}$ is denoted by $\mathsf{A} \cdot \mathsf{int}_j$. Moreover, automata accepting all possible behaviors over any given set of subformulae is denoted by Σ^*. Moreover,

$$\exists c'.Seg^+(\rho', i'_{k-1}, c, S') \in L(\mathsf{A}_k[init, q] \cdot \mathsf{int}_j) \iff Seg^+(\rho', i'_{k-1}, m', S') \in L(\mathsf{A}_k[init, q] \cdot \mathsf{int}_j \cdot \Sigma^*). \tag{19}$$

Hence, by Equations (17), (18), and (19), we have $\rho', i' \models \phi'_k \iff$

$$\rho', i' \models \bigvee_{j=0}^{\mathsf{cmax}-1} \bigvee_{q \in Q} \bigvee_{f \in F} \Bigg[\Bigg\{ \exists i' \le i'_1 \le \dots \le i'_{k-1}. \bigwedge_{g=1}^{k-1} (\tau'_{i'_g} - \tau'_{i'}) \in \mathsf{I}_g \wedge \rho', i'_g \models \bigvee \Sigma$$
$$\wedge \mathrm{Seg}^+(\rho', i'_{g-1}, i'_g, S') \in L(\mathsf{A}'_g)) \wedge Seg^+(\rho', i'_{k-1}, m', S') \in L(\mathsf{A}_k[init, q] \cdot \mathsf{int}_j \cdot \Sigma^*)\Bigg\}$$
$$\wedge \Bigg\{\exists i'_k.(\tau'_{i'_k} - \tau'_{i'}) \in \mathsf{I}_k.\rho', i'_k \models \psi^-(k, q, f, j) \wedge \mathrm{Seg}^+(\rho', i'_k, m', S') \in L(\mathsf{A}'_{k+1})\Bigg\}\Bigg]. \tag{20}$$

By semantics of PnEMTL and EMITL logic, the above condition is equivalent to $\rho', i' \models \psi^{1.1}_f =$

$$\bigvee_{j=0}^{\mathsf{cmax}-1} \bigvee_{q \in Q} \bigvee_{f \in F} [\{\mathcal{F}^{k-1}_{\mathsf{I}_1,\dots,\mathsf{I}_{k-1}}(\mathsf{A}'_1,\dots,\mathsf{A}'_{k-1}, \mathsf{A}_k[init, q] \cdot \mathsf{int}_j \cdot \Sigma^*)\} \wedge \{\mathsf{F}_{I_k}(\psi^-(k, q, f, j) \wedge \mathcal{F}(\mathsf{A}'_{k+1} \cdot \mathsf{Last}))\}]. \tag{21}$$

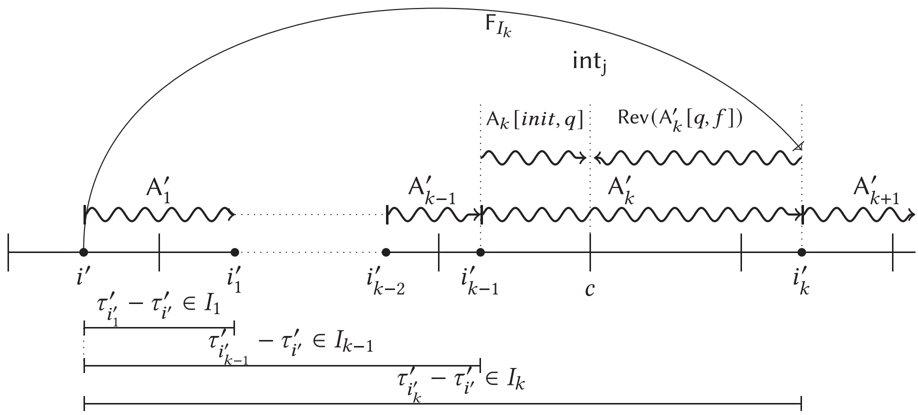

Fig. 10. Figure showing reduction of k-ary PnEMTL formulae to $k-1$-ary PnEMTL formulae when i'_{k-1} and i'_k are not within same integer time points (Case 1.1). The behavior from i' to nearest oversampling point c after i'_{k-1} (labelled int_j) is given by $\mathcal{F}_{I_1,\ldots,I_{k-1}}(\mathsf{A}'_1,\ldots,\mathsf{A}_k[init,q]\cdot\mathsf{int}_j.\Sigma^*)$, and the corresponding behavior from c is given by $\{\mathsf{F}_{I_k}(\mathcal{P}(\mathsf{Rev}(\mathsf{A}'_k[q,f])\cdot\mathsf{int}_j))\wedge\mathcal{F}(\mathsf{A}_{k+1}\cdot\mathsf{Last})\}$, where q is any state of A'_k reached when read till int_j and f is the final state that is reached when i'_k is read. Disjuncting over all possible (but finitely many) $j\in\{0,\ldots,\mathsf{cmax}-1\}$, $q\in Q_i$ and $f\in F_i$, we get the required formulae.

- **Case 1.2:** I_k is unbounded, i.e., $u=\infty$. There are two possibilities. Either (1) i'_k occurs within $\langle l_k, l_k+1)$ from i' or (2) i'_k occurs beyond timestamp $\lceil\tau'_{i'}+l_k\rceil$:
  Possibility 1: i'_k occurs within $\langle l_k, l_k+1)$. Hence, $\tau'_{i'_k}\in\langle\tau'_{i'}+l_k,\tau'_{i'}+l_k+1)$. The case can be handled using the following formulae:
  $$\phi^{1.2.1}=\mathcal{F}^{k-1}_{\mathsf{I}_1,\ldots,\mathsf{I}_{k-1},\langle l_k,l_k+1)}(\mathsf{A}_1,\ldots,\mathsf{A}_{k+1}). \tag{22}$$
  This formulae now falls under case 1.1, as I_k and I_{k-1} are non-overlapping and bounded and, $l_k+1\le\mathsf{cmax}$. Hence, it can be handled similarly. Let $\psi^{1.2.1}$ be the formula that we get after applying the reduction as mentioned in case 1.1 on $\phi1.2.1_f$.
  Possibility 2: $\tau'_{i'_k}>\lceil\tau'_{i'}+l_k\rceil$. In this case, we use the oversampling point, c, at integer timestamp $\lceil\tau'_{i'}+l_k\rceil$ to break the check for A_k. Note that if the nearest oversampling integer point next to i' is labelled int_j, then (by φ_{ovs}), $\lceil\tau'_{i'}+l_k\rceil$ is labelled as $\mathsf{int}_{j'}$ where $j'=j\oplus l_k$. Moreover, as the time difference between $\mathsf{int}_{j'}$ and i' is less than $l_k+1\le\mathsf{cmax}$, there is no other oversampling point between i' and c (hence, between i'_{k-1} and c) with the same label as $\mathsf{int}_{j'}$. Hence, to assert A_k from i'_{k-1} to some point beyond c, we start with checking for A'_k from i'_k to c reaching some state q and continuing the run from c to check for the remaining part starting from q and ending up at some final state f at some point (i.e., the required i'') in the future of c from where we again assert that the behavior till the end of the word is accepted by A'_{k+1} (in which case i'' becomes the required i'_k). This can be expressed using the following formula:

$$\psi^{1.2.2}=\bigvee_{j=0}^{\mathsf{cmax}-1}\left[(\mathsf{F}_{[0,1)}\mathsf{int}_j)\rightarrow\left\{\bigvee_{q\in Q}\bigvee_{f\in F}[\mathcal{F}^{k-1}_{\mathsf{I}_1,\ldots,\mathsf{I}_{k-1}}(\mathsf{A}_1,\ldots,\mathsf{A}_{k-1},\mathsf{A}'_k[init,q]\cdot\mathsf{int}_{j'},\mathsf{A}'_k[q,f]\cdot\mathsf{A}_{k+1})\right]\right\}. \tag{23}$$

We encourage readers to refer to Figure 11. Hence, $\psi^{1.2}=\psi^{1.2.1}\vee\psi^{1.2.2}$ is the required formula for this case. Note that Possibilities 1 and 2 are not disjoint. That is, there are positions of i'_k that fall within both sets of possibilities. This simply means that there are models for which both

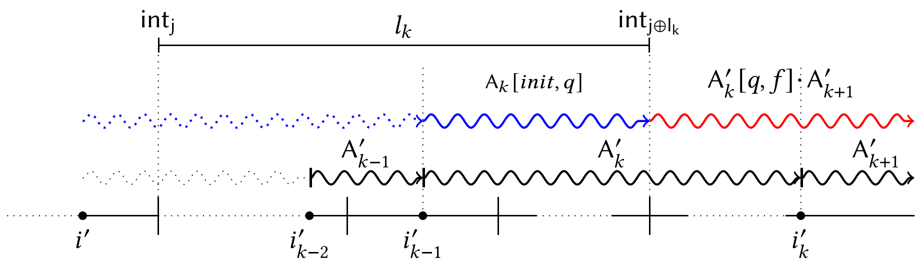

Fig. 11. Figure showing reduction of k-ary PnEMTL formulae to $k-1$-ary PnEMTL formulae when the intervals I_{k-1} and I_k are non-overlapping but I_k is unbounded (Case 1.2). In this case, we will not be able to make sure that time difference between i'_{k-1} and i'_k is bounded by cmax. This figure highlights the reduction for only one of the possibilities (i'_k occurs after time $\lceil \tau'_{i'} + l_k \rceil$ and hence beyond l_k time from i') of this case and hence gives one of the disjunct for required formula. The behavior from i' to oversampling point at time $\lceil \tau'_{i'} + l_k \rceil$ (say, c) is given by the part of the following formula colored blue $\mathcal{F}_{I_1,\ldots,I_{k-1}}(A'_1,\ldots,A'_k[init,q]\cdot \mathsf{int}_{j'}, A'_k[q,f]\cdot A'_{k+1})$ and the part beyond c is given by the part of the formula colored in red where $j' = j \oplus l_k$, q is any state of A'_k reached when read till c and f is the final state that is reached when i'_k is read. Disjuncting over all possible (but finitely many) $j \in \{0,\ldots,\mathrm{cmax}-1\}$, $q \in Q_i$ and $f \in F_i$, we get the required formulae.

$\psi^{1.2.1}, \psi^{1.2.2}$ hold. Hence, the restrictions imposed by both these formulae might be redundant, but together both these formulae cover all the possibilities for occurrence of i'_k.

- **Case 2:** I_{k-1} and I_k overlap each other. That is, $l_k < u_{k-1}$ (again, the strict $<$ is due to the fact that ϕ'_f is a non-adjacent formula). Hence, it is possible that there is no oversampling point between i'_{k-1} and i'_k, because of which we can not only rely on Lemma 8.3. There are following subcases depending on how the intervals I_{k-1} and I_k overlap and whether I_k is bounded or not:
  - **Case 2.1:** I_k is bounded, $l_{k-1} = l_k$ and $u_{k-1} < u_k$. There are two possibilities based on the relative positions of i'_{k-1} and i'_k.
  Possibility 1: There is an oversampling point between i'_{k-1} and i'_k. As I_k is bounded, the time difference between the former and the latter is bounded by cmax. Hence, using Lemma 8.3 and identical reasoning used in case 1.1, the same formula $\psi^{1.1}$ takes care of this possibility.
  Possibility 2: There is no oversampling point between i'_{k-1} and i'_k. If i'_{k-1} lies within I_{k-1} from i', then $\tau'_{i'_k} \le \lceil \tau'_{i'_k} \rceil = \lceil \tau'_{i'_{k-1}} \rceil$(timestamps of both the points have same integer parts) $< \tau'_{i'_{k-1}} + 1$(property of ceiling function) $\le \tau'_{i'} + 1 + u_{k-1}$ (i'_{k-1} lies within I_{k-1} of i') $\le \tau'_{i'} + u_k$ ($u_k < u_{k-1}$ and u_k is an integer). Similarly, i'_{k-1} lies within I_{k-1}, and the i'_k occurs after i'_{k-1} implies the time difference between i'_k and i' is more than $l_{k-1} = l_k$ units. Hence, (a) i'_{k-1} within I_{k-1} from i', (b) i'_k occurs after i'_{k-1}, and (c) there is no oversampling point between them, implies that i'_k is within I_k from i'. We check this inline using the following $k-1$ ary PnEMTL formula: (a) is checked using the last interval I_{k-1}, (b) is asserted by concatenating A_k with A'_{k+1} appearing in the last argument, and (c) is asserted by using A_k (which disallows any oversampling points) rather than A'_k for concatenation with A'_{k+1} in the last argument. Hence, the following formula covers this possibility:

$$\psi^{2.1.2} = \mathcal{F}^{k-1}_{I_1,\ldots I_{k-1}}(A'_1,\ldots,A'_{k-1}, A_k \cdot A'_{k+1}). \tag{24}$$

  We encourage the readers to go through Figure 12. Finally, the formula for this kind of overlapping of I_{k-1} and I_k is $\psi^{2.1} = \psi^{1.1} \vee \psi^{2.1.2}$.

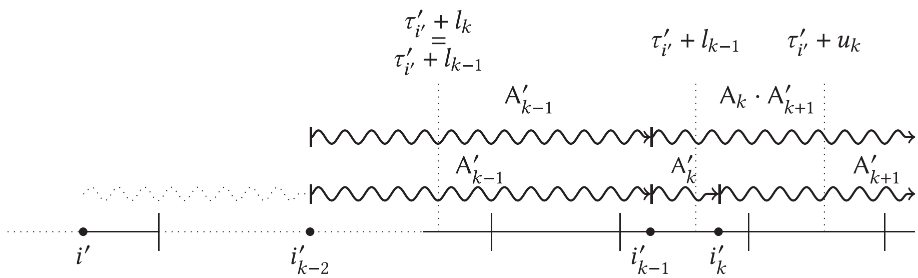

Fig. 12. Figure showing reduction of k-ary PnEMTL formulae to $k-1$-ary PnEMTL formulae when the intervals I_{k-1} and I_k are overlapping but I_k is bounded $l_k = l_{k-1}$ and $u_k > u_{k-1}$ (Case 2.1). Hence, in this case, we will not be able to make sure that there is an oversampling point between i_{k-1} and i_k. This diagram covers the situation where there is no oversampling point between i_{k-} and i_k. For the situation where there is an oversampling point between these points, we can use the formula identical to case 1.1. If i'_{k-1} is within I_{k-1} and i'_k occurs after that but before the next oversampling point (that is, the integer part of their timestamps are same), then i_k is within I_k. Hence, we just need to check that from i'_{k-1} there exists a point in the future c before the next oversampling point such that the behavior from i'_{k-1} till that point is given by A_k rather than A'_k (as the former disallows occurrence of all oversampling points) and from c onwards A'_{k+1} holds till the last point of the word. Then, c is a valid candidate for the required i'_k. Hence, we remove the interval I_k from the given formula and replace the last argument A'_{k+1} by $\mathsf{A}_k \cdot \mathsf{A}'_{k+1}$.

– **Case 2.2:** I_k is bounded, $u_{k-1} = u_k$ and $l_{k-1} < l_k$. This case is similar to the case above. There are two possibilities as in case 2.1.
Possibility 1: There is an oversampling point between i'_{k-1} and i'_k. Similar to case 1.1 and 2.1, $\psi^{1.1}$ covers this possibility.
Possibility 2: There is no oversampling point between i'_{k-1} and i'_k. The argument here is symmetric. But we just need to check i'_k's position rather than i'_{k-1}. If i'_k lies within I_k from i', then $\tau'_{i'_{k-1}} \geq \lfloor \tau'_{i'_k} \rfloor$ (timestamps of both the points have same integer parts) $> \tau'_{i'_k} - 1$ (property of floor function) $\geq \tau'_{i'} - 1 + l_k$ (i'_k lies within I_k of i') $\leq \tau'_{i'} + l_{k-1}$ ($l_{k-1} < l_k$ and l_{k-1} is an integer). Similarly, i'_k lies within I_k and the i'_{k-1} occurs before i'_k implies the time difference between i'_{k-1} and i' is less than $u_k = u_{k-1}$ units. Hence, (a) i'_k occurring within I_k from i', (b) i'_{k-1} occurs after i'_k, and (c) there is no oversampling point between them, implies that i'_{k-1} is within I_{k-1} from i'. We check this inline using the following $k-1$-ary PnEMTL formula: (a) is checked setting the last interval I_k, (b) is asserted by concatenating A'_{k-1} with A_k in the second last argument, and (c) is asserted by using A_k (which disallows any oversampling points) rather than A'_k for concatenation with A'_{k-1} in the last but second argument. Hence, the following formula covers this possibility:

$$\psi^{2.2.2} = \mathcal{F}^{k-1}_{I_1,\ldots,I_{k-2},I_k}(\mathsf{A}'_1, \ldots, \mathsf{A}'_{k-1} \cdot \mathsf{A}_k, \mathsf{A}'_{k+1}). \tag{25}$$

Finally, the formula for this kind of overlapping between I_{k-1} and I_k is $\psi^{2.2} = \psi^{1.1} \vee \psi^{2.2.2}$.
– **Case 2.3:** I_k is bounded, $l_{k-1} < l_k$ and $u_{k-1} < u_k$. As above, there are two possibilities. Possibility 1, when point i'_{k-1} and i'_k have an oversampling point in between them. This possibility is identical to case 1.1 and hence $\psi^{1.1}$ covers it. For possibility 2, when both the points are within same integer timestamps, consider the following: Let $I' = I_{k-1} \cap I_k = \langle l_k, u_{k-1} \rangle$.[12] Then, for possibility 2 to occur, either (a) i'_{k-1} occurs within I' from i' or (b) i'_k occur within I' from i', because if both of them do not occur within the intersection, then

[12]The left bracket will depend on interval I_k and the right will depend on I_{k-1}.

there is at least one oversampling point between them (which is already covered by $\psi^{1.1}$)). Hence, it suffices to reduce arity of formula $\phi^{2.3} = \phi_a^{2.3} \vee \phi_b^{2.3}$ for this possibility where

$$\phi_a^{2.3} = \mathcal{F}^{k-1}_{I_1,\ldots,I_{k-2},I',I_k}(\mathsf{A}'_1,\ldots,\mathsf{A}'_{k-1},\mathsf{A}'_k,\mathsf{A}'_{k+1}), \tag{26}$$

$$\phi_b^{2.3} = \mathcal{F}^{k-1}_{I_1,\ldots,I_{k-2},I_{k-1},I'}(\mathsf{A}'_1,\ldots,\mathsf{A}'_{k-1},\mathsf{A}'_k,\mathsf{A}'_{k+1}). \tag{27}$$

Note that these k-ary PnEMTL formulae can be reduced individually, as $\phi_a^{2.3}$ falls under the case 2.1, while $\phi_b^{2.3}$ falls under the case 2.2. Let $\psi_a^{2.3}$ and $\psi_b^{2.3}$ be the formulae we get after applying the reduction from cases 2.1 and 2.2 to the formulae $\phi_a^{2.3}$ and $\phi_b^{2.3}$, respectively. Then, the required formula covering this case is $\psi^{2.3} = \psi^{1.1} \vee \psi_a^{2.3} \vee \psi_b^{2.3}$.

– **Case 2.4:** $l_{k-1} = l_k = l$ and $u_{k-1} = u_k = u$. Let $I_{k-1} = (l, u) = I_k$.[13] Like the previous subcases, possibility in which there is an oversampling point between i'_k and i'_{k-1}, is handled by formula $\psi^{1.1}$. There are two other possibilities.

Possibility 2: (a) There is no oversampling point between i'_{k-1} and i'_k, and (b)$\tau'_{i'_{k-1}} \in (\tau'_{i'} + l, \lfloor \tau'_{i'} + u \rfloor)$. Note that (a) and (b) implies i'_{k-1} and i'_k are within (l, u) from i'. To check (a), we nest a $\mathcal{F}$ modality within the $k-1$-ary PnEMTL formula asserting A_k instead of A'_k from point i'_{k-1} in the kth argument of $k-1$-ary PnEMTL formula (see formula Γ_j). To check (e), we just have to assert that if $\rho', i' \models \mathsf{F}_{[0,1)}(\mathsf{int}_j)$, then $\tau'_{i'_{k-1}} - \tau'_{i'} \in (l, u) \wedge \rho', i'_{k-1} \models \neg\mathsf{F}_{[0,1)}(\mathsf{int}_{j\oplus u_k})$ (again, see formula Γ_j). Let $\Gamma_j = \neg\mathsf{F}_{[0,1)}(\mathsf{int}_{j\oplus u_k}) \wedge \mathcal{F}(\mathsf{A}_k \cdot \mathsf{A}'_{k+1})$ and $S'' = S' \cup \Gamma_j$. Let $\mathcal{S}$ sets of subsets of S'' containing Γ_j.

$$\psi^{2.4(i)} = \bigvee_{X \in \mathcal{S}} \subseteq \bigvee_{j=1}^{\mathsf{cmax}} (\mathsf{F}_{[0,1)}(\mathsf{int}_j) \rightarrow \mathcal{F}^{k-1}_{I_1,\ldots,I_{k-1}}(\mathsf{A}''_1,\ldots,\mathsf{A}''_{k-1}, X \cdot \Sigma^*)(S''), \tag{28}$$

where $\mathsf{A}''_i = \mathsf{A}_i'^{\Gamma_j}$. That is, the transitions of A''_i do not depend on the truth value of Γ_j.

Possibility 3: (a) holds and (c) $\tau'_{i'_k} \in (\lfloor \tau'_{i'} + u_k \rfloor, \tau'_{i'} + u_k)$. Then, (a) and (c) implies i'_{k-1} and i'_k are within (l, u) from i'. Like the previous possibility, to check (a), concatenate A'_{k-1} with A_k instead of A'_k in the $k-1$th argument of $k-1$-ary PnEMTL formula. To check (c), we just have to assert that if $\rho', i' \models \mathsf{F}[0,1)(\mathsf{int}_j)$, then $\tau'_{i'_k} - \tau'_{i'} \in (l, u) \wedge \rho', i'_{k-1} \models \mathsf{F}_{[0,1)}(\mathsf{int}_{j\oplus u_k})$ (check Γ'_j). Let $\Gamma'_j = \mathsf{F}_{[0,1)}(\mathsf{int}_{j\oplus u_k}) \wedge \mathcal{F}(\mathsf{A}'_{k+1})$, $S'' = S' \cup \Gamma'_j$. Let $\mathcal{S}'$ sets of subsets of S'' containing Γ'_j.

$$\psi^{2.4(ii)} = \bigvee_{X \in \mathcal{S}'} \wedge\Gamma_j \in S \subset \bigvee_{j=1}^{\mathsf{cmax}} (\mathsf{F}_{[0,1)}(\mathsf{int}_j) \rightarrow \mathcal{F}^{k-1}_{I_1,\ldots,I_{k-2},I_k}(\mathsf{A}''_1,\ldots,\mathsf{A}''_{k-1} \cdot \mathsf{A}_k^{\Gamma'_j}, X \cdot \Sigma^*), \tag{29}$$

where $\mathsf{A}''_i = \mathsf{A}_i'^{\Gamma'_j}$ for $i \leq k-1$. Let $\psi^{2.4} = \psi^{1.1} \vee \psi^{2.4(i)} \vee \psi^{2.4(ii)}$.

• **Case 2.5:** I_k is an unbounded interval. We break this case into two possibilities. (1) i'_k occurs within $J_1 = \langle l_k, u_{k-1}+1)$.[14] (2) i'_k occurs within $J_2 = [u_{k-1}+1, \infty)$. Hence, ϕ'_f can be rewritten as $\phi'_{f,1} \vee \phi'_{f,2}$ where for $i \in \{1, 2\}$,

$$\phi'_{f,i} = \mathcal{F}^k_{I_1,\ldots,I_{k-1},J_i}(\mathsf{A}'_1,\ldots,\mathsf{A}'_{k-1},\mathsf{A}'_k,\mathsf{A}'_{k+1}). \tag{30}$$

---

[13]The proof can be extended to handle other kinds of intervals similarly.

[14]$u_{k-1} + 1 \leq \mathsf{cmax}$.

Note that $\phi'_{f,1}$ falls under case 2.3 if $l_k > l_{k-1}$ and case 2.1 if $l_k = l_{k-1}$. Moreover, $\phi'_{f,2}$ is within the case 1.2 and, hence, can be handled accordingly. Let $\psi'_{f,1}$ and $\psi'_{f,2}$ be the formulae we get after applying corresponding reductions to $\phi'_{f,1}$ and $\phi'_{f,2}$, respectively. Then, $\psi^{2.5} = \psi'_{f,1} \vee \psi'_{f,2}$.

- Note that all other cases are disallowed by Assumptions 1 and 2.

Hence, the required formula ϕ_f^{k-1} depends on the type of intervals I_k and I_{k-1}. For example, if I_k is bounded and does not have an intersection with I_{k-1}, then it falls within case 1.1 and $\psi^{1.1}$ is the required PnEMTL. Moreover, note that the total number of operators (temporal, Boolean, etc.) in PnEMTL and EMITL$_{0,\infty}$ is $O(\mathsf{cmax} \times |Q_k| \times |F_k| \times |\phi_f|)$. □

After recursively applying the above reductions, we get a formula ψ_f that contains modalities from EMITL$_{0,\infty}$ and F_{np} of the size in $O((\mathsf{cmax} \times |\phi_f|)^n) = O(|\phi_f|^{Poly(n,M)})$, where n is the arity of ϕ_f and M is the number of bits required to store the constants of its timing interval.

8.4 Eliminating $\mathsf{F}_{\langle l,u \rangle}$ Modalities Where $l > 0$ And $u \neq \infty$

Let $I = \langle l, u \rangle$ be any interval appearing in ψ' where $l > 1$ and $u < \infty$ (hence, $u \leq \mathsf{cmax}$). Let $\rho' = (\sigma'_1, \tau'_1) \dots (\sigma'_{m'}, \tau'_{m'})$ such that $\rho' \models \varphi_{ovs}$. In this section, given any formula of the form $\mathsf{F}_I(\alpha)$, we construct a specification $\delta(I, \alpha)$ using α and modalities from MITL$_{0,\infty}$ such that, for any $i' \in dom(\rho')$, $\rho', i' \models \mathsf{F}_I(\alpha)$ iff $\rho', i' \models \delta(\mathsf{F}_I, \alpha)$. Notice that $\rho', i' \models \mathsf{F}_I(\alpha)$ iff there exists a point $j > i'$ such that $\tau'_j - \tau'_{i'} \in \langle l, u \rangle$ and $\rho', j \models \alpha$. Let c be the nearest integer oversampling point after i'. Let c' be the integer oversampling point with timestamp $\lceil \tau'_{i'} + l \rceil$. There are two possibilities, depending on the occurrence of j.

Case 1: Either $\tau'_j \in \langle \tau'_{i'} + l, \lceil \tau'_{i'} + l \rceil]$. This implies j occurs before c'. If int_j is true at point c, then $\mathsf{int}_{j \oplus l}$ is true at point c'. Moreover, c' will be the very first point after i' with $\mathsf{int}_{j \oplus l}$ counter value as $l \leq \mathsf{cmax}$. Hence, if j is any point in $\langle l, \infty)$ from i' that occurs before c, then timestamp of j is within $\langle \tau'_{i'} + l, \lceil \tau'_{i'} + l \rceil]$. This could be easily expressed using formula $\delta_1(I, \alpha) = \bigvee_{i=0}^{\mathsf{cmax}-1} [\mathsf{F}_{[0,1)}(\mathsf{int}_j) \rightarrow (\neg \mathsf{int}_{j \oplus l} \mathsf{U}_{\langle l, \infty)} \alpha)]$.

Case 2: Or $\tau'_j \in (\lceil \tau'_{i'} + l \rceil, \tau'_{i'} + u \rangle$. Hence, j occurs after c'. Notice that u is not greater than cmax. Hence, all the oversampling points in $\langle \tau'_{i'}, \tau'_{i'} + u \rangle$ are labelled with unique counters as the counters increment modulo cmax. Hence, the counter values at all the oversampling points between c and c' are different than the counter values (or labels) at all the oversampling points with timestamp in $\tau'_{i'} + l \rceil, \tau'_{i'} + u \rangle$. More precisely, if c is labelled with int_j, then all the oversampling points within $(\tau'_{i'}, \lceil \tau'_{i'} + l \rceil]$ will be labelled with propositions in $\{\mathsf{int}_{j \oplus 1}, \dots \mathsf{int}_{j \oplus l}\}$, while the oversampling points within $(\lceil \tau'_{i'} + l \rceil, \lceil \tau'_{i'} + u \rceil \rangle$ will be labelled with propositions from $E = \{\mathsf{int}_{j \oplus l+1}, \dots, \mathsf{int}_{j \oplus u}\}$. Hence, any point within $[0, u\rangle$ of i' where $\mathsf{F}_{[0,1)} \bigvee E$ holds, occurs within time $(\lceil \tau'_{i'} + l \rceil, \tau'_{i'} + u \rangle$. Hence, to assert that $j \in (\lceil \tau'_{i'} + l \rceil, \tau'_{i'} + u \rangle$, we construct formula $\delta_2(I, \alpha) = \bigvee_{i=0}^{\mathsf{cmax}-1} [\mathsf{F}_{[0,1)}(\mathsf{int}_j) \rightarrow \mathsf{F}_{[0,u)}(\alpha \wedge \mathsf{F}_{[0,1)}(\bigvee E))]$.

Hence, $\delta(I, \alpha) = \delta_1(I, \alpha) \vee \delta_2(I, \alpha)$ is the required formula free from any bounded interval with non-zero lower bound (provided α is free from such intervals). Notice that size of $\delta(I, \alpha) = O(\mathsf{cmax} \times |\mathsf{F}_I(\alpha)|)$.

Hence, when this step is applied to formula ψ_f from the previous step, we get a formula ψ'_f in EMITL$_{0,\infty}$ (MITL$_{0,\infty}$ is a sublogic of EMITL$_{0,\infty}$). Moreover, in the DAG corresponding to ψ_f there will be at most $|\psi_f|$ F_{np} operators. Hence, size of ψ'_f is $O(|\psi_f| \times \mathsf{cmax}) = O(|\phi_f|^{Poly(n,M)})$.

Applying all the above steps to every PnEMTL modality in ϕ_{flat}, we get a formula $\psi' \in$ EMITL$_{0,\infty}$ that is equisatisfiable to ϕ. Moreover, the size of ψ' is at most $|\psi|$ times the size of max ψ'_f. Hence, ψ' is of the size in $O(|\phi|^{Poly(n,M)})$.

8.5 Concluding Proof

The above three steps of construction show that:

- The equisatisfiable $\mathrm{EMITL}_{0,\infty}$ formula ψ is of the size $(O(|\phi|^{Poly(n,M)})$, where n is the arity ϕ and M is the number of bits required to store the constants appearing in the timing intervals. This is because reducing the arity of each subformula ϕ_f by 1 results in formula of $O(\mathsf{cmax}\times |\phi_f|^3)$ size. Hence, after recursively applying the reduction to get the final $\mathrm{EMITL}_{0,\infty} + \mathsf{F}_{np}$ formula, we get a formula of the $O((\mathsf{cmax} \times |\phi|^3)^n)$ $(|\phi| > |\phi_f|)$ size. Eliminating F_{np} will blow up the size by cmax. (i) Hence, the required formula is of $O(\mathsf{cmax} \times (\mathsf{cmax} \times |\phi|^3)^n)$ size. As $\mathsf{cmax} = 2^M$ (M defined above), the size of the final required formula is bounded by $O(2^{Poly(n,M)})$.
- For a non-adjacent 1-TPTL formula γ, applying the reduction in Section 5 yields ϕ of size (ii) $O(2^{Poly|\gamma|})$ and, arity of $\phi = O(|\gamma|^2)$ and the set of constants remain the same. Note the set of constants used in the timing interval of output formula ϕ is the same as that of γ. Hence, the number of bits required to store the constants in ϕ is (iii) $M = O(|\gamma|)$. Also, after applying the reduction of Section 8 by plugging the value of $|\phi|$ and its arity from (ii) and value of M from (iii) in (i), we get the $\mathrm{EMITL}_{0,\infty}$ formula ψ of size $O(2^{Poly(|\gamma|)*Poly(n,M)}) = O(2^{Poly(|\gamma|)})$.
- By Lemma 7.3, given any formula γ in NA-GQMSO, we can construct an equivalent formula ϕ in NA-PnEMTL (with non-elementary blow-up) that can then be analyzed for satisfaction, as presented above. Hence, satisfiability for NA-GQMSO is decidable. Non-elementary lower bound for NA-GQMSO is inherited by the subclass FO[<].

9 A NOTE ON INFINITE TIMED WORDS

Up until this point, we have restricted our models to be finite timed words. Let Σ be any finite set of propositions. An infinite or ω-timed word over Σ is an infinite sequence of the form $(\sigma_1, \tau_1)(\sigma_2, \tau_2)\ldots$ where $\forall \in \mathbb{N}$, σ_i is a non-empty subset of Σ, $\tau_1 = 0$ and $i, j \in \mathbb{N}$ $i < j$ implies $\tau_i \leq \tau_j$. An ω-timed word is said to be zeno if the limit of the sequence $\tau_1, \tau_2, \ldots,$ is not infinite. This means there are infinite actions within a finite duration. For example, $(a, 0)(a, 0.5)(a, 0.75)(a, 0.875)\ldots$ is a zeno timed word. It is a common practice in the literature to restrict the models to non-zeno words, as physical systems do not exhibit zeno behavior: It would take infinite amount of energy to carry out infinitely many actions in finite time. Hence, we restrict ourselves to non-zeno models.[15] The set of all non-zeno ω-timed words over Σ is denoted by $T\Sigma^\omega$. On closer inspection, it can be seen that all the results for finite timed words in the previous sections can be easily lifted to infinite timed words. We point out the required modifications for this lifting. In the rest of this section, let $\rho = (\sigma_1, \tau_1)\ldots$ be any non-zeno ω-timed word. Let A be any Büchi Automata. Let $L^\omega(\mathrm{A})$ denote the set all untimed ω-words accepted by A.

9.1 Definition of Logics over Infinite Timed Words

The syntax and semantics for logic LTL, MTL, TPTL, and MSO remain the same. For EMTL and PnEMTL, the following changes are required:

9.1.1 EMTL Extended with Büchi Automata Modalities. We extend EMTL with a new modality, $\mathcal{F}^\omega(\mathrm{A})(S)$, where A is a Büchi Automata modality over subformulae S. Intuitively, this modality asserts that from the given point the suffix is accepted by A. Let $S = \{\varphi_1, \ldots, \varphi_n\}$. For any $x \in \mathbb{N}$, let S_x be the exact subset of formulae in S that holds at point x of ρ. Then, for any $i \in \mathbb{N}$, $\mathrm{Seg}^\omega(\rho, i, S)$ is an untimed ω-word $S_i S_{i+1} \ldots$. Define $\rho, i \models \mathcal{F}^\omega(\mathrm{A})(S)$ iff $\mathrm{Seg}^\omega(\rho, i, S) \in L^\omega(\mathrm{A})$.

[15]That is, language of any formula ϕ can only contain non-zeno timed words.

9.1.2 PnEMTL Extended with Büchi Automata Modalities. Syntactically, in the modality $\mathcal{F}^k(\mathsf{A}_1, \ldots, \mathsf{A}_{k+1})$, A_{k+1} is a Büchi Automata, while the rest $\mathsf{A}_1, \ldots, \mathsf{A}_k$ are classical non-deterministic automata with reachability objective. The new semantics of the $\mathcal{F}^k_{\mathsf{I}_1,\ldots,\mathsf{I}_k}$ modality is as follows:

- $\rho, i_0 \models \mathcal{F}^k_{I_1,\ldots,I_k}(\mathsf{A}_1, \ldots, \mathsf{A}_{k+1})(S)$ iff $\exists i_0 \leq i_1 \leq i_2 \ldots \leq i_k$ s.t. $\bigwedge_{w=1}^{k} [(\tau_{i_w} - \tau_{i_0} \in I_w) \wedge \mathsf{Seg}^+(\rho, i_{w-1}, i_w, S) \in L(\mathsf{A}_w)] \wedge \mathsf{Seg}^\omega(\rho, i_k, S) \in L^\omega(\mathsf{A}_{k+1})$.

Intuitively, as the suffix from i_k onwards is infinite, it is natural to check the behavior in that region by a Büchi Automaton. The syntax and semantics of the $\mathcal{P}^k_{\mathsf{I}_1,\ldots,\mathsf{I}_k}$ modality does not change.

9.2 Anchored Interval Infinite Timed Words

As the name suggests, Anchored Interval ω-words are ω extension of interval words. For the sake of completeness, we define these formally. Let $\mathcal{I}_\nu \subseteq \mathcal{I}_{\mathsf{int}}$. An $\mathcal{I}_\nu$ anchored ω-interval word over Σ is an ω-word κ of the form $\sigma_1\sigma_2 \ldots \in 2^{\Sigma \cup \{anch\} \cup \mathcal{I}_\nu}$ such that there is a unique point $i \in \mathbb{N}$ where anch holds. As before, this point is called an anchor point of κ and denoted by $\mathsf{anch}(\kappa)$. Moreover, for every $i \in \mathbb{N}$, $\Sigma \cap \sigma_i \neq \emptyset$. That is, at every point in κ, at least one of the propositions from Σ holds. Let $\rho = (\sigma_1, \tau_1) \ldots$ be any ω-timed word. ρ, i is consistent with an I_ν ω-interval word $\kappa = \sigma'_1 \ldots$ if and only if for any $j \in \mathbb{N}$, $\sigma'_j \cap \Sigma = \sigma_j$, $\tau_j - \tau_i \in \sigma'_j \cap \mathcal{I}_\nu$ and $i = \mathsf{anch}(\kappa)$. Let $\mathsf{Time}(\kappa)$ be all the non-zeno pointed ω-timed word ρ, i consistent with κ. In what follows, let κ be an $\mathcal{I}_\nu$ ω-interval word. Let $I \in \mathcal{I}_\nu$ be any interval of the form $\langle l, u\rangle$, where $u \neq \infty$. If $\{j | I \in \kappa[j]\}$ is an infinite set (i.e., I occurs infinitely often in κ), then we call κ a *Zeno Interval Word.* The following proposition is straightforward:

Proposition 9.1. *If κ is a Zeno Interval Word and if ρ, i is consistent with κ, then ρ is a zeno word.*

This is because there will be infinitely many points j is $\rho = (\sigma_1, \tau_1) \ldots$ such that $\tau_j - \tau_i \leq u$. This, by definition, implies that ρ is a zeno word. Hence, if κ is a Zeno Interval Word, then $\mathsf{Time}(\kappa)$ is an empty set. The definition of Collapsed ω-Interval word is the same as Collapsed Interval words appearing in Section 4. As the proof of Lemma 4.2 does not require κ to be a finite word, it holds for non-zeno ω-interval words, too.

9.2.1 Normalization. For a collapsed non-zeno ω-interval word κ and $I \in \mathcal{I}_\nu$, let $\mathsf{first}(\kappa, I)$ and $\mathsf{last}(\kappa, I)$ denote the positions of first and last occurrence of I (as defined in Section 4). If I occurs infinitely often, then $\mathsf{last}(\kappa, I)$ is undefined. $\mathsf{Norm}(\kappa) = \sigma'_1\sigma'_2 \ldots$ is an $\mathcal{I}_\nu$ ω-interval word built from κ as follows:

- **Reduction 1:** For every unbounded interval $I \in I\nu$, delete all the occurrences of I except the first one. Let this be denoted as $R_1(\kappa)$
- **Reduction 2:** For every unbounded interval $I \in I\nu$, delete all the occurrences of I except the first and the last one.

For any non-zeno word κ, unbounded interval $I = \langle l, \infty) \in \mathcal{I}_\nu$ and $x \in \mathbb{N}$, $x = \mathsf{first}(\kappa, I)$ implies for all $\rho, i \in \mathsf{Time}(\kappa)$ and $y > x$, $\tau_y - \tau_i \in I$. Hence, any occurrence of interval I after its first occurrence is redundant, as the same restriction is imposed by the first occurrence of I. Hence, we have the following proposition:

Proposition 9.2. *For any non-zeno interval word κ, if κ' is obtained from κ by applying Reduction 1 defined above, then $\kappa \cong \kappa'$.*

Hence, for any collapsed word non-zeno κ, $\kappa' = R_1(\kappa)$ will contain only finitely many time-restricted points. As every interval $\mathcal{I} \in \mathcal{I}_\nu$ appears finitely often in κ', Lemma 4.4 is now applicable for κ'. Hence, by Lemmas 4.2, 4.4, and 9.2, we have, for any non-zeno word κ, $\kappa \cong \mathsf{Norm}(\kappa)$.

9.3 Translation from 1-TPTL to PnEMTL

All the reduction in Section 5.1.1, i.e., translation from simple TPTL formulae to LTL over interval words, remains the same, as all the lemmas in that section hold for non-zeno infinite words, too. Translation from LTL to Büchi Automata over Collapsed Interval Words remains the same, as those techniques and results are standard for both finite and infinite timed words.

9.3.1 Partitioning of Interval Words. While the general idea of partitioning the Language of NFA over the interval words into finitely many type sequences remains the same, we need to make some changes to the construction from A to Aut_{seq} to incorporate Reduction 1 of normalization of ω-interval words. In particular, we need to make sure that the $Status(I_j)$ for any unbounded interval $I_j \in \mathcal{I}_\nu$ does not change from *mid* to *post*. This is because we essentially want to erase all the occurrences of I_j after the first one. Hence, *Choice*1 transition is deleted for unbounded intervals in Aut_{seq}.

9.3.2 Reducing NFA of Each Type to PnEMTL. Section 5.3 remains the same. All the automata A_1 to A_k are automata over finite words (as the intervals only appear within the finite prefix of accepted words). Automaton A_{k+1} is a Büchi Automaton. Hence, in the PnEMTL formulae, the last argument will be a Büchi Automaton, which is in agreement with the new syntax and semantics introduced in Section 9.1.

9.4 Equivalence of PnEMTL and GQMSO

Here, too, the existing reduction from PnEMTL to GQMSO and vice versa works. The only difference is, we need to use the standard Büchi Elgot Trakhtenbrot Theorem for infinite words.

9.5 Satisfiability Checking for Non-adjacent PnEMTL

This remains the same, too. The only difference is, wherever A_{n+1} appears, it is a Büchi Automaton. As in the reduction mentioned in Section 8, A_{n+1} always appears as the last argument (or the tail automaton) in the modalities of EMITL and PnEMTL in output and all the intermediate formulae. This is in accordance with the new syntax and semantics of PnEMTL for infinite words, as mentioned in Section 9.1. And in the base case Lemma 8.5, A'_2 is a Büchi Automaton. We reduce the logic to $\mathrm{EMITL}_{0,\infty}$ extended with $\mathcal{F}^\omega$. On inspection of Reference [27], the $\mathcal{F}^\omega$ modality could be trivially reduced to Büchi Timed Automaton with size polynomial to that of the formulae. Hence, the result.

10 CONCLUSION

We generalized the notion of non-punctuality to non-adjacency in logics TPTL and GQMSO. We proved that satisfiability checking for the non-adjacent 1-variable fragment of TPTL is EXPSPACE Complete. This gives us a strictly more expressive logic than MITL while retaining the satisfiability complexity. We introduced a new logic called PnEMTL and used it to solve the satisfiability checking problem for both non-adjacent 1-TPTL and GQMSO. The added expressive power over MITL comes with a useful ability to specify complex sequences of timing constraints over regular behaviors (automata). All our results, including decidability, extend to infinite timed words, as outlined in Section 9.

We believe that our logics and decidability results are useful for the specification and design of real-time systems. In model-based temporal planning, timing constraints on events are specified using logical formulae. Satisfiability checking of such formulae return a model that essentially gives a schedule meeting all the planning constraints. Several papers have investigated the use of TPTL with past modalities in formulating time-line-based planning [9–11, 21, 36]. Our expressive logics subsume several of these, and they offer a possibility of modelling even more general timing constraints involving regular behaviors (see the example below). The satisfiability checking method of this article potentially gives us a technique for automatic synthesis of plans. In another line of work investigating top-down design of real-time systems, assumptions and commitments over real-time systems are specified in a real-time logic. Moreover, design decisions (in the form of desired constraints on the behavior of the system to be implemented) can also be encoded in logic. Verification of this design step involves showing that the commitment is logically implied by the conjunction of assumptions and design decisions (see References [13, 38] for early examples of this approach). Validity checking of logical formulae (equivalently, satisfiability checking of negated formulae) permits automatic verification of such design decisions. However, an experimental validation of usefulness of these methods for practical planning and verification remains to be investigated.

Example 10.1. Consider a job (e.g., automated pizza-maker) containing some high-level activities (involving several sub-steps) given by a sequence of finite state automata $P_1, P_2, \ldots P_k$ (e.g., kneading the dough, preheating oven, baking the pizza) that has to be performed in a given sequence atomically (i.e., without pre-emption). Each process P_k has a deadline u_k associated with it. Now, we need to plan these processes such that the job is successfully completed within m time units. Moreover, there are some extra restrictions specified by, for instance, an MITL formula φ. This could be done by finding a finite word satisfying the following formula:

$$\phi = \mathcal{F}^k_{[0,u_1),\ldots,[0,u_k)}(P_1, P_2, \ldots, P_k, \mathsf{Finish}) \wedge \varphi.$$

In case of GQMSO, the fact that the alternation of metric quantifiers in an anchored block can be eliminated using extra non-metric quantifiers (see Theorem 6.4) is an interesting result, in our opinion.

Finally, we pose the following open problems that we believe are fundamentally interesting and worth solving:

- **Is non-adjacent 1-TPTL strictly more expressive than MITL with Pnueli modalities (and hence Q2MLO of Reference [26])?** We conjecture a positive answer to the question. More precisely, we conjecture that the property "within interval $(1, 2)$ from the present point, events a and b occur such that a is immediately followed by event b" is not expressible using MITL with Pnueli modalities and hence in Q2MLO. But this is easily expressible in non-adjacent 1-TPTL and hence in GQFO (first-order fragment of GQMSO) as follows: $x.\mathsf{F}(a \wedge T - x \in (1, 2) \wedge \oplus(b \wedge T - x \in (1, 2)))$.
- **How does the logic non-adjacent GQMSO compare with the class of two-way deterministic timed automata with reversal boundedness [3] and MIDL [17]?** Is there any natural subclass of timed automata corresponding to GQMSO? If yes, then it will be the largest known subclass of timed automata (to the best of our knowledge) that is closed under complementation. Ferrère in Reference [17] gives a very elegant extension of MITL called **Metric Interval Dynamic Logic (MIDL)**, where the timing constraints are associated with regular expressions (Metric Interval Regular Expressions) as opposed to the modalities. While EMITL is a syntactic subclass of both non-adjacent PnEMTL and MIDL of Reference [17], there are still gaps in the expressiveness relationships amongst these logics. Ferrère already proved that MIDL is strictly more expressive than EMITL with only future automata modalities. However, (i) is EMITL with past modalities strictly included in non-adjacent PnEMTL and (ii) how non-adjacent PnEMTL compares with MIDL of Reference [17] (if you allow/disallow past operators in both logics) in terms of expressiveness is still open.
- **Efficient Tool Development for NA-1-TPTL Satisfiability and Model Checking**. While we show that the satisfiability checking problem for NA-1-TPTL is in EXPSPACE, the

algorithm is merely a proof-of-concept. We believe that in spite of the inherent worst-case theoretical complexity of the problem, in practice, we can build scalable tools for automated verification of NA-1-TPTL properties. Our decidability proof relies on the reduction of any NA-1-TPTL formula to an equisatisfiable $\mathrm{EMITL}_{0,\infty}$ formula. Using techniques similar to References [32] and [35], we can reduce $\mathrm{EMITL}_{0,\infty}$ formulae to equisatisfiable MITL formulae. This can then be followed by using scalable tools, such as MightyL [12] (automata-based tool) and Reference [7] (SMT-based tool) for MITL satisfiability and model checking. Using the reduction by Ho in Reference [27], we can also reduce this $\mathrm{EMITL}_{0,\infty}$ formula to an equivalent timed automata, which can be analyzed using scalable tools such as UPPAAL [6] and TChecker [24]. A direct simpler reduction from the satisfiability checking problem of NA-1-TPTL is also an intriguing open problem.

- We also leave open an exploration for a suitable definition for non-adjacency and its satisfiability checking problem in the context of TPTL with multiple variables.

ACKNOWLEDGMENTS

We thank the reviewers of FM 2021 and the journal *Formal Aspects of Computing* for their feedback to improve the article.

REFERENCES

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

[2] Rajeev Alur, Tomás Feder, and Thomas A. Henzinger. 1991. The benefits of relaxing punctuality. In *10th Annual ACM Symposium on Principles of Distributed Computing*. ACM, 139–152. DOI: https://doi.org/10.1145/112600.112613

[3] Rajeev Alur and Thomas A. Henzinger. 1992. Back to the future: Towards a theory of timed regular languages. In *33rd Annual Symposium on Foundations of Computer Science*. IEEE Computer Society, 177–186. DOI: https://doi.org/10.1109/SFCS.1992.267774

[4] Rajeev Alur and Thomas A. Henzinger. 1993. Real-time logics: Complexity and expressiveness. *Inf. Comput.* 104, 1 (1993), 35–77. DOI: https://doi.org/10.1006/inco.1993.1025

[5] Rajeev Alur and Thomas A. Henzinger. 1994. A really temporal logic. *J. ACM* 41, 1 (Jan. 1994), 181–203. DOI: https://doi.org/10.1145/174644.174651

[6] Johan Bengtsson, Kim Guldstrand Larsen, Fredrik Larsson, Paul Pettersson, and Wang Yi. 1995. UPPAAL—A tool suite for automatic verification of real-time systems. In *DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems (Lecture Notes in Computer Science, Vol. 1066)*. Springer, 232–243. DOI: https://doi.org/10.1007/BFb0020949

[7] Marcello M. Bersani, Matteo Rossi, and Pierluigi San Pietro. 2015. An SMT-based approach to satisfiability checking of MITL. *Inf. Computat.* 245 (2015), 72–97. DOI: https://doi.org/10.1016/j.ic.2015.06.007

[8] Patricia Bouyer, Fabrice Chevalier, and Nicolas Markey. 2005. On the expressiveness of TPTL and MTL. In *Conference on Foundations of Software Technology and Theoretical Computer Science*. Springer Berlin, 432–443.

[9] Laura Bozzelli, Alberto Molinari, Angelo Montanari, and Adriano Peron. 2019. Model checking timeline-based systems over dense temporal domains. In *20th Italian Conference on Theoretical Computer Science (CEUR Workshop Proceedings, Vol. 2504)*. CEUR-WS.org, 235–247. Retrieved from http://ceur-ws.org/Vol-2504/paper27.pdf.

[10] Laura Bozzelli, Alberto Molinari, Angelo Montanari, Adriano Peron, and Gerhard J. Woeginger. 2020. Timeline-based planning over dense temporal domains. *Theor. Comput. Sci.* 813 (2020), 305–326. DOI: https://doi.org/10.1016/j.tcs.2019.12.030

[11] Laura Bozzelli, Angelo Montanari, and Adriano Peron. 2019. Taming the complexity of timeline-based planning over dense temporal domains. In *39th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 150)*. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 34:1–34:14. DOI: https://doi.org/10.4230/LIPIcs.FSTTCS.2019.34

[12] Thomas Brihaye, Gilles Geeraerts, Hsi-Ming Ho, and Benjamin Monmege. 2017. MightyL: A compositional translation from MITL to timed automata. In *29th International Conference on Computer Aided Verification (Lecture Notes in Computer Science, Vol. 10426)*. Springer, 421–440. DOI: https://doi.org/10.1007/978-3-319-63387-9_21

[13] Zhou Chaochen, Charles Anthony Richard Hoare, and Anders P. Ravn. 1991. A calculus of durations. *Inf. Process. Lett.* 40, 5 (1991), 269–276.

[14] Stéphane Demri, Valentin Goranko, and Martin Lange. 2016. *Temporal Logics in Computer Science: Finite-state Systems*, Vol. 58. Cambridge University Press.

[15] Cindy Eisner and Dana Fisman. 2006. *A Practical Introduction to PSL*. Springer.

[16] Calvin C. Elgot. 1961. Decision problems of finite automata design and related arithmetics. *Trans. Amer. Math. Soc.* 98 (1961), 21–51.

[17] Thomas Ferrère. 2018. The compound interest in relaxing punctuality. In *22nd International Symposium on Formal Methods, Held as Part of the Federated Logic Conference (Lecture Notes in Computer Science, Vol. 10951)*. Springer, 147–164. DOI: https://doi.org/10.1007/978-3-319-95582-7_9

[18] IEEE P1850-Standard for PSL-Property Specification Language. 2005. https://ieeexplore.ieee.org/document/4040498.

[19] Carlo A. Furia and Matteo Rossi. 2008. MTL with bounded variability: Decidability and complexity. In *Formal Modeling and Analysis of Timed Systems*. Springer Berlin, 109–123.

[20] Paul Gastin and Denis Oddoux. 2003. LTL with past and two-way very-weak alternating automata. In *28th International Symposium on Mathematical Foundations of Computer Science (Lecture Notes in Computer Science, Vol. 2747)*. Springer, 439–448. DOI: https://doi.org/10.1007/978-3-540-45138-9_38

[21] Nicola Gigante. 2019. Timeline-based planning: Expressiveness and complexity. *CoRR* abs/1902.06123 (2019).

[22] Christoph Haase, Joël Ouaknine, and James Worrell. 2010. On process-algebraic extensions of metric temporal logic. In *Reflections on the Work of C. A. R. Hoare*. Springer, 283–300. DOI: https://doi.org/10.1007/978-1-84882-912-1_13

[23] Thomas A. Henzinger, Jean-François Raskin, and Pierre-Yves Schobbens. 1998. The regular real-time languages. In *25th International Colloquium on Automata, Languages and Programming (Lecture Notes in Computer Science, Vol. 1443)*. Springer, 580–591. DOI: https://doi.org/10.1007/BFb0055086

[24] Frédéric Herbreteau, B. Srivathsan, and Igor Walukiewicz. 2016. Better abstractions for timed automata. *Inf. Comput.* 251 (2016), 67–90. DOI: https://doi.org/10.1016/j.ic.2016.07.004

[25] Y. Hirshfeld and A. Rabinovich. 2006. An expressive temporal logic for real time. In *Mathematical Foundations of Computer Science Conference*. 492–504.

[26] Yoram Hirshfeld and Alexander Rabinovich. 2006. Expressiveness of metric modalities for continuous time. In *Computer Science – Theory and Applications*. Springer Berlin, 211–220.

[27] Hsi-Ming Ho. 2019. Revisiting timed logics with automata modalities. In *22nd ACM International Conference on Hybrid Systems: Computation and Control*. ACM, 67–76. DOI: https://doi.org/10.1145/3302504.3311818

[28] P. Hunter. 2013. When is metric temporal logic expressively complete? In *Computer Science Logic (CSL'13)*, 380–394. https://drops.dagstuhl.de/opus/volltexte/2013/4209/.

[29] Specification IEEE Standard for SystemVerilog: Unified Hardware Design and Verification Language. 2005. *IEEE Std 1800-2005* (2005), 1–648. DOI: https://doi.org/10.1109/IEEESTD.2005.97972

[30] J. R. Büchi. 1962. On a Decision Method in Restricted Second-order Arithmetic. https://link.springer.com/chapter/10.1007/978-1-4613-8928-6_23#citeas.

[31] S. N. Krishna, K. Madnani, and P. K. Pandya. 2014. Partially punctual metric temporal logic is decidable. In *International Workshop on Temporal Representation and Reasoning*. 174–183.

[32] Shankara Narayanan Krishna, Khushraj Madnani, and Paritosh K. Pandya. 2017. Making metric temporal logic rational. In *42nd International Symposium on Mathematical Foundations of Computer Science (LIPIcs, Vol. 83)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 77:1–77:14. DOI: https://doi.org/10.4230/LIPIcs.MFCS.2017.77

[33] Shankara Narayanan Krishna, Khushraj Madnani, and Paritosh K. Pandya. 2018. Logics meet 1-clock alternating timed automata. In *29th International Conference on Concurrency Theory (LIPIcs, Vol. 118)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 39:1–39:17. DOI: https://doi.org/10.4230/LIPIcs.CONCUR.2018.39

[34] Slawomir Lasota and Igor Walukiewicz. 2008. Alternating timed automata. *ACM Trans. Comput. Log.* 9, 2 (2008), 10:1–10:27. DOI: https://doi.org/10.1145/1342991.1342994

[35] Khushraj Nanik Madnani. 2019. *On Decidable Extensions of Metric Temporal Logic*. Ph. D. Dissertation. Indian Institute of Technology Bombay, Mumbai, India.

[36] Dario Della Monica, Nicola Gigante, Angelo Montanari, Pietro Sala, and Guido Sciavicco. 2017. Bounded timed propositional temporal logic with past captures timeline-based planning with bounded constraints. In *26th International Joint Conference on Artificial Intelligence*. ijcai.org, 1008–1014. DOI: https://doi.org/10.24963/ijcai.2017/140

[37] J. Ouaknine and J. Worrell. 2005. On the decidability of metric temporal logic. In *Annual Symposium on Logic in Computer Science*. 188–197.

[38] Paritosh K. Pandya. 2001. Specifying and deciding quantified discrete-time duration calculus formulae using DCVALID. In *RTTOOLS2001 Workshop*. BRICS. Retrieved from https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.30.8532.

[39] Paritosh K. Pandya and Simoni S. Shah. 2011. On expressive powers of timed logics: Comparing boundedness, nonpunctuality, and deterministic freezing. In *22nd International Conference on Concurrency Theory (Lecture Notes in Computer Science, Vol. 6901)*. Springer, 60–75. DOI: https://doi.org/10.1007/978-3-642-23217-6_5

[40] Pavithra Prabhakar and Deepak D'Souza. 2006. On the expressiveness of MTL with past operators. In *International Conference on Formal Modeling and Analysis of Timed Systems*. 322–336.

[41] A. Rabinovich. 2008. Complexity of metric temporal logic with counting and Pnueli modalities. In *International Conference on Formal Modeling and Analysis of Timed Systems.* 93–108.

[42] Alexander Rabinovich. 2010. Complexity of metric temporal logics with counting and the Pnueli modalities. *Theor. Comput. Sci.* 411, 22-24 (2010), 2331–2342. DOI: https://doi.org/10.1016/j.tcs.2010.03.017

[43] Jean Francois Raskin. 1999. *Logics, Automata and Classical Theories for Deciding Real Time.* Ph. D. Dissertation. Universite de Namur.

[44] B. A. Trakhtenbrot. 1961. Finite automata and logic of monadic predicates. *Doklady Akad. Nauk SSSR, in Russian.* 98 (1961), 326–329.

[45] Moshe Y. Vardi and Pierre Wolper. 1994. Reasoning about infinite computations. *Inf. Computat.* 115, 1 (1994), 1–37.

[46] Thomas Wilke. 1994. Specifying timed state sequences in powerful decidable logics and timed automata. In *3rd International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems, Organized Jointly with the Working Group Provably Correct Systems.* 694–715. DOI: https://doi.org/10.1007/3-540-58468-4_191

Received 28 April 2022; revised 21 September 2022; accepted 31 October 2022