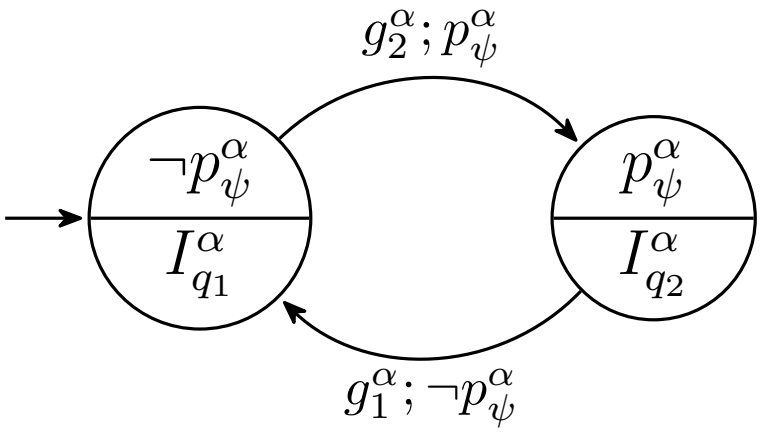

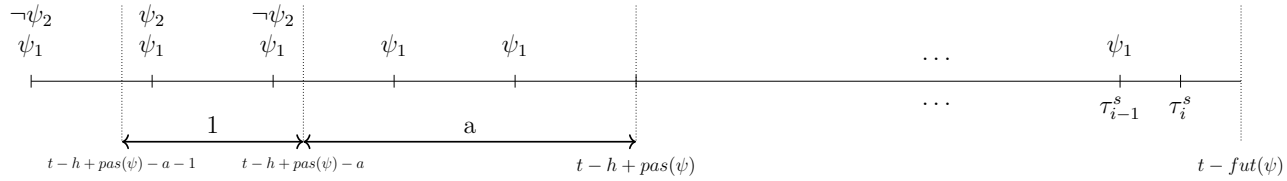

Translating point-in-time Metric Temporal Logic to Timed Automata

Master's thesis in Computer science and engineering

Mattis Eeten-Jeppsson

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

Master's thesis 2021

Translating point-in-time Metric Temporal Logic to Timed Automata

Mattis Eeten-Jeppsson

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

Translating point-in-time Metric Temporal Logic to Timed Automata
Mattis Eeten-Jeppsson

Supervisor: Nir Piterman, Department of Computer Science and Engineering
Examiner: Gerardo Schneider, Department of Computer Science and Engineering

Master's Thesis 2021
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Description of the picture on the cover page (if applicable)

Typeset in LaTeX
Gothenburg, Sweden 2021

Translating point-in-time Metric Temporal Logic to Timed Automata
MATTIS EETEN-JEPPSSON
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Metric Temporal Logic (MTL) extends the modalities of LTL with timing, making it well suited for specifying properties over real timed traces. This thesis is concerned with applying MTL to runtime verification with the purpose of obtaining a monitoring procedure based on timed automata. Target semantics are stated which are weak and strong truncated paths semantics for point-in-time MTL and which give a verdict for every finite or infinite trace. While these semantics clearly state our intention they are impractical for our purposes and we give equivalent unified semantics based on extending finite paths with "all true" and "all false". This is the first such treatment given for the point-in-time domain. An existing construction based on timed automata is then adapted to implement the target semantics and is extended with the past fragment of MTL. Finally it is shown how this construction can be applied to monitoring.

Keywords: computer science, automata theory, logic, runtime verification, metric temporal logic, timed automata.

Acknowledgments

First I want to thank my examiner Gerardo Schneider for valuable feedback and comments that improved the quality and readability of this thesis.

I cannot express enough thanks to my supervisor Nir Piterman for his continued support throughout my work on this thesis. He always took time to help me think through my ideas, usually with some valuable insight.

I want to thank my parents and my brother for their encouragement.

Finally, I want to thank my wife Lisa for always being there for me and for giving me the strength to continue until the finish line. Thank you for being in my life.

Mattis Eeten-Jeppsson, Troisdorf, June 2021

Contents

List of Figures

Chapter 1

Introduction

Software verification has the goal of ensuring programs conform to their stated requirements. Theorem proving [11], model checking [13] and testing [24] are the methods of verification that have traditionally been considered in the literature, but all of these impose constraints on the software development process. In contrast *runtime verification* [22][4] is a light-weight form of verification which deals only with the detection of violations of the specification for a program at runtime. Compared to other methods it requires only a specification and a way to collect traces and may thus enable use of verification methods in contexts where it was previously deemed too costly or not infeasible due to limitations imposed by other methods.

In runtime verification a trace of events is generated from a running program and the trace is given to a monitoring procedure which verifies that the trace conforms to a given specification for the program. In this model every event in the trace consists of the the set of *atomic propositions* that were true when the event occurred and trace is either *accepted* or *rejected* according to the specification. An important distinction is made between *online monitoring* in which a monitoring procedure is given events one at a time and directly gives a verdict for the prefix of events observed so far following each event, and *offline monitoring* in which a monitoring procedure is given an entire trace and gives a verdict for the entire trace.

The specification is typically expressed at a high level of abstraction using temporal logic, with Linear Temporal Logic (LTL) [27] being the canonical example. While LTL has modalities for expressing properties in the past and future, it lacks a more general concept of time, limiting its ability to specify when in time such properties are meant to occur.

Metric Temporal Logic (MTL) [20] extends the modalities of LTL with timing. The main difference between MTL and LTL is that the past and future modalities in MTL, $\mathcal{S}$(ince) and $\mathcal{U}$(ntil) have associated (relative) time intervals specifying

how far into the past or future properties are to occur. MTL comes in continuous time and point-in-time variations. In the former case the input is a signal w, a function that to every time $t \in \mathbb{R}_{\geq 0}$ assigns a set of atomic propositions, and in the latter it is a timed word s which can be thought of as a timestamped sequence of events.

The semantics of LTL as given in [27] and of MTL in [20] define acceptance for infinite traces. Traces from running programs are however necessarily finite and thus runtime verification requires alternative semantics. To accommodate this various semantics for LTL over finite traces have been proposed, including *truncated path semantics* [14] and *three-valued semantics* [8], and such semantics can typically be restated for MTL in a straightforward manner.

Connections between automata theory [19] and temporal logic have been studied extensively and questions in temporal logic can often be answered by restating them as corresponding automata theoretic questions. For LTL translations to automata over infinite traces, so called ω-automata, are since long part of the standard tool set [31] and such translations have been used in runtime monitoring, for example in Bauer et al. [6]. Alur and Dill [2] introduced *timed automata* which are ω-automata over timed words and signals in order to model real-time system. Translations from MTL and its relative *Metric Interval Temporal Logic* (MITL) to timed automata exist, for example Ničković and Piterman [25] for MTL and Maler et al. [23] for MITL but we are not aware of any monitoring procedure for MTL based on timed automata despite numerous algorithms based on other techniques, such as dynamic programming and various untimed ω-automata. The research that gets closest to such a procedure is Bauer et al. [6, 8], but they consider TLTL, a logic distinct from MTL.

1.1 Problem statement

Given the strong connections between temporal logics and automata theory, and the existence of timed automata, which themselves are defined over the same domain as MTL, namely timed words and signals, it is surprising that translations from MTL to timed automata have not been used more for runtime monitoring. There are many theoretical advantages to using an automata based approach, with the existence of underlying theory and decision procedures being two of them.

In this thesis we intend to remedy the situation somewhat by beginning an exploration into runtime monitoring of MTL specifications using timed automata. More concretely this thesis is concerned with applying MTL to runtime verification with the purpose of obtaining a monitoring procedure based on timed automata which would be feasible to implement and use.

We propose to use an existing translation from MTL to timed automata by Ničković

and Piterman [25]. It has the advantage, although we do not make use of it here, of being fully determinizable, meaning that the resulting non-deterministic timed automata can be converted to deterministic timed automata. The continuous-time semantics they implement is however over infinite traces and thus cannot be directly used for runtime monitoring. In addition many programs eligible for runtime monitoring are of a more discrete nature although steps may be taken with arbitrary timing and thus using point-in-time semantics seems more appropriate. Finally many temporal properties are easier to express using a mixture of past and future operators but [25] only implements the future fragment of MTL.

The requirements mentioned in the previous paragraph are all related to semantics. A suitable variation of semantics for finite traces must be chosen. Out of the two main contenders we pick truncated path semantics [14]. These semantics can be defined inductively which makes them easier to reason about compared to the three-valued semantics of [6] which do not have this advantage [7]. In runtime verification performance is important but the criteria for rejection in three-valued semantics, namely recognition of *bad prefixes*, requires an ω-automaton with number of states doubly exponential in the size of the formula while truncated-path semantics only suffers a singly exponential blowup [21].

Efficiency is important in runtime verification, especially in the case of online monitoring where the monitoring procedure runs alongside the system under test, but also in the general case having a more efficient procedure widens its applicability, for example by enabling larger specifications to be monitored. The translation in Ničković and Piterman [25] translates bounded subformulas and other constraints into first-order predicates and relies on quantifier elimination, a procedure that in the worst case increases the size of the constraint by an exponential factor. Our choice of point-in-time semantics should remove the need for quantifier elimination by replacing quantification by finite conjunctions and disjunctions of propositions.

With these prerequisites out of the way we should be able to derive a monitoring procedure for point-in-time MTL over truncated paths by translating into timed automata.

1.2 Research objectives

These are the hypotheses the thesis will try to confirm:

H1 Ničković and Piterman [25] can be adapted to point-in-time MTL with truncated paths semantics and past fragment and optimized so that it does not suffer the exponential blowup of quantifier elimination..

H2 Given such a construction it is possible to derive a feasible procedure for runtime monitoring.

1.3 Methodology

We begin by stating our target semantics. These are point-in-time MTL with weak and strong truncated path semantics, extended to include the past fragment of MTL in addition to the future fragment. We then give equivalent unified semantics, based on extending finite paths with "all true" and "all false". This formulation is more appropriate for our purposes as it gives a single inductively defined entailment relation rather than two.

With the semantics in place we proceed to the translation to timed automata. We use the translation from MTL to timed automata by Ničković and Piterman [25] as a starting point, but as that construction is for continuous time MTL over infinite traces, and uses timed automata that read signals, rather than timed words, it cannot be directly used for point-in-time MTL with truncated path semantics. We can however translate a timed word into a signal in a relatively straightforward manner by letting propositions keep their last truth value at every point in time between two events. In addition we add a special proposition τ which holds precisely at the times when an event occurred in the timed word. The point-in-time semantics require us to memorize the times that events occurred and in addition the propositions that were true for each event. We add a timed automaton for memorizing events and adapt a timed automaton for memorizing propositions from [25].

Events have arbitrary timing, although the number of events in an interval are assumed to be bounded, and in order for a timed automaton to be able memorize them we alter the definition of the timed automata used to allow instantaneous transitions when special proposition τ is carried by the signal.

A big challenge is creating predicates for the weak and strong semantics. This challenge is mainly due the weak semantics and the strong semantics respectively being defined as each other's duals under negation. We solve this by defining two sets of predicates, each representing the truth of subformulas located at a depth with given parity relative to the top formula with respect to negation. This solution is similar in spirit to how one might solve the corresponding problem in alternating automata translations for LTL, for example [12]. These predicates are stated as finite conjunction and disjunctions of proposition and do not use quantifiers.

In addition we are adding the past fragment of MTL. For this we provide a timed automaton which is able to memorize the truth of subformulas in the past, and as expected for the past this automaton is completely deterministic.

Significant effort goes into proving the correctness of the translation, namely

that a timed word is accepted by the resulting timed automaton iff it is accepted by the formula under the target semantics.

Finally we give a monitoring procedure. This is achieved by reducing the monitoring problem to checking a specially constructed timed automata for emptiness.

1.4 Thesis overview

The remainder of this thesis is organized in the following manner: In Chapter 2 we state our target semantics, which are weak and strong truncated paths semantics [14] for point-in-time MTL. These semantics give a verdict for every finite or infinite trace. While these semantics clearly state our intention they are impractical for our purposes, and we give equivalent unified semantics based on the $\top, \bot$ approach [15]. This is the first such treatment given for the point-in-time domain. Chapter 3 gives background on timed automata. In Chapter 4 the construction from [25] is adapted to implement the target semantics and this includes adding the past fragment of MTL and proving equivalence between the construction and the target semantics. We also sketch a monitoring procedure based on the construction. In Chapter 5 we briefly survey the relevant literature and its relation to our work. Finally, in Chapter 6 we present our conclusions and suggest possible directions for future work.

Chapter 2

Point-in-time semantics

The previous chapter briefly introduced runtime verification as a way of verifying that a program conforms to a given specification, and temporal logics such as LTL as powerful tools for expressing such specifications. It was noted that LTL lacks a concept of time, making it less suited for specifying properties over real-time traces, but that its derivative MTL fills this gap. Finally it was stated that the goal of the thesis is obtaining a monitoring procedure based on translating point-in-time MTL formulas to timed automata.

Before we are ready to proceed to translating MTL to timed automata it is necessary to make precise what semantics will be used. The topic of this chapter is defining a set of semantics that are relevant for our goal of monitoring without over complicating implementation. Semantics that provide verdicts for traces of arbitrary length are a requirement for runtime verification. Section 2.1 gives *weak* and *strong* semantics which meet this requirement, but the fact that they are stated as two relations makes them less suited for implementation. Section 2.2 remedies this by giving equivalent semantics as a single relation.

2.1 Weak and strong semantics

A reasonable assumption in runtime verification is that the trace is a timed word s, making point-in-time semantics applicable. In general an MTL formula may require an infinite word to decide satisfaction, but traces collected from running programs are necessarily finite. A monitoring procedure for a formula φ when given a trace s must thus determine whether s satisfies φ, does not satisfy φ, or s is too short to reach a conclusion. For this purpose a *truncated paths* interpretation of MTL is used. This interpretation gives *weak* and *strong* views to the satisfaction of a formula, with weak satisfaction intuitively corresponding to nothing bad happening, and strong satisfaction corresponding to all expectations being met.

We give weak and strong semantics for MTL with past and future modalities over timed words, extending that of [14], who gave semantics for LTL.

The syntax of an MTL formula is defined inductively by:

$$\varphi := \ p \mid \neg\varphi_1 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \mathcal{U}_I \varphi_2 \mid \varphi_1 \mathcal{S}_I \varphi_2$$

where p belongs to the set of atomic propositions $AP = \{p_1, \ldots, p_m\}$, φ_1 and φ_2 are subformulas and I is one of (a, b), $[b, b]$ or (a, ∞), where a and b are non-negative integers and $a < b$. Addition of an interval I with a scalar $x \in \mathbb{R}$ is defined as $(a, b) + x = (a + x, b + x)$, $(a, \infty) + x = (a + x, \infty)$, and $[b, b] + x = [b + x, b + x]$. The negation $-I$ of I is defined as $-(a, b) = (-b, -a)$, $-(a, \infty) = (-\infty, -a)$, $-[b, b] = [-b, -b]$. Finally we define derived operators $\Diamond_I \psi \stackrel{def}{=} \text{true} \mathcal{U}_I \psi$ and $\Box_I \psi \stackrel{def}{=} \neg(\Diamond_I \neg\psi)$.

Formally, a timed word $s = (\rho^s, \tau^s)$ is a pair of indexed sequences, ρ^s and τ^s, of identical length $|s|$ with $\rho^s_i \in 2^{AP}$ denoting the atomic propositions that are true for the ith event and $\tau^s_i \in \mathbb{R}^+$ denoting the time of the ith event. Every pair of times of τ^s satisfy $\tau^s_i < \tau^s_j$ if $i < j$. The concatenation $s' \cdot s''$ of two timed words s' and s'' is the timed word s such that $\rho^s_i = \rho^{s'}_i$ and $\tau^s_i = \tau^{s'}_i$ if $i \leq |s'|$ and $\rho^s_i = \rho^{s''}_{i-|s'|}$ and $\tau^s_i = \tau^s_{|s|} + \tau^{s''}_{i-|s'|}$ if $i > |s'|$.

With these preliminaries we define *weak* and *strong semantics*:

Definition 1. *Weak semantics are defined by:*

- $s, i \models^- p$ *iff* $i > |s|$ *or* $p \in \rho^s_i$.
- $s, i \models^- \neg\varphi$ *iff* $s, i \not\models^+ \varphi$.
- $s, i \models^- \varphi_1 \vee \varphi_2$ *iff* $s, i \models^- \varphi_1$ *or* $s, i \models^- \varphi_2$.
- $s, i \models^- \varphi_1 \mathcal{U}_I \varphi_2$ *iff one of:*
  - *There is* $j > i$, $j \leq |s|$ *and* $\tau^s_j - \tau^s_i \in I$, *such that* $s, j \models^- \varphi_2$ *and for every* k, $i < k < j$ $s, k \models^- \varphi_1$.
  - $i > |s|$.
  - $\tau^s_{|s|} - \tau^s_i < sup(I)$ *and for every* k, $i < k \leq |s|$ $s, k \models^- \varphi_1$.
- $s, i \models^- \varphi_1 \mathcal{S}_I \varphi_2$ *iff either:*
  - $i > |s|$.
  - *There is* j, $1 \leq j < i$ *with* $\tau^s_i - \tau^s_j \in I$ *such that* $s, j \models^- \varphi_2$ *and for every* k, $j < k < i$ $s, k \models^- \varphi_1$.

Strong semantics are defined by:

- $s, i \models^+ p$ *iff* $i \leq |s|$ *and* $p \in \rho_i^s$.
- $s, i \models^+ \neg\varphi$ *iff* $s, i \not\models^- \varphi$.
- $s, i \models^+ \varphi_1 \vee \varphi_2$ *iff* $s, i \models^+ \varphi_1$ *or* $s, i \models^+ \varphi_2$.
- $s, i \models^+ \varphi_1 \mathcal{U}_I \varphi_2$ *iff there is* j, $i < j \leq |s|$, *with* $\tau_j^s - \tau_i^s \in I$ *such that* $s, j \models^+ \varphi_2$ *and for every* k, $i < k < j$ $s, k \models^+ \varphi_1$.
- $s, i \models^+ \varphi_1 \mathcal{S}_I \varphi_2$ *iff* $i \leq |s|$ *and there is* $1 \leq j < i$ *with* $\tau_i^s - \tau_j^s \in I$ *such that* $s, j \models^+ \varphi_2$ *and for every* k, $j < k < i$ $s, k \models^+ \varphi_1$.

These are the semantics our monitoring procedure should implement.

Some comments about these semantics are warranted. We first give some intuition of the meaning of acceptance and rejection under the weak and strong semantics respectively, and then proceed with explaining how these verdicts are used for reasoning about finite traces.

The challenge in giving verdicts to finite traces lies mainly in deciding how to treat obligations in the future that are not met because the trace is too short. For example, the formula $\varphi = p\mathcal{U}_{(0,10)}q$ promises that there is some event in the next ten time units where q holds, and additionally that p holds for every event between then and the current event. If the last event of s is more than ten time units later than the current event it can easily be determined whether these obligations were met by inspecting the trace.

If on the other hand the last event of s is less than ten time units later than the current event and there are still outstanding obligations (that is q has yet to hold and it cannot be ruled out that some event could appear where p does not hold before that event) a judgment must be made based on incomplete information. In this case the weak semantics will accept s as long as there is no event in s directly contradicting φ. Conversely, the strong semantics will reject s as long as not every obligation in φ is met by events in s

Another example of this difference in judgment by the two semantics is $\varphi = p$ where $s, i \models^- p$ and $s, i \not\models^+ p$ if $i > |s|$. A somewhat unintuitive consequence is that $s, i \models^- p \wedge \neg p$ and $s, i \not\models^+ p \vee \neg p$ if $i > |s|$. In addition there is no finite s such that $s, i \models^- \neg\Diamond_{(0,\infty)}p$ or $s, i \not\models^+ \Box_{(0,\infty)}p$.

The weak and strong semantics as defined above have the strength relation $s, i \models^+ \varphi$ implies $s, i \models^- \varphi$ (and consequently $s, i \not\models^- \varphi$ implies $s, i \not\models^+ \varphi$). For purposes of runtime verification the verdicts given by the weak and strong semantics can be combined to distinguish three different outcomes (summarized in Figure 2.1) of a run of the program being tested, namely a successful where every obligation was met, detection of a direct violation of a requirement and thus a defect in the program, or finally the trace being to short to reach a conclusion.

	weak reject	weak accept
strong reject	defect detected in program	trace too short
strong accept		successful test

Figure 2.1: The conclusion drawn from different outcomes of weak and strong semantics. Note that the case weak rejection and strong acceptance is not possible because of the strength relation of $\models^+$ and $\models^-$

2.2 $\top, \bot$ semantics

The weak and strong semantics as presented in Section 2.1 are our target semantics, but the fact that they are defined by two separate relations is impractical for our purposes, not the least because of duplication. In this section we show that the weak and strong semantics can be expressed in a unified way if the alphabet of the timed words is extended with special symbols $\top$ and $\bot$, following the approach for LTL given in [15].

The possible events of ρ^s are extended with symbols $\top$ and $\bot$ so that $\rho^s_i \in \{\top, \bot\} \cup 2^{AP}$. The complement of s, $\overline{s}$ is obtained by replacing all occurrences of $\top$ in ρ^s_i by $\bot$, and all occurrences of $\bot$ with $\top$, respectively.

With this we define:

Definition 2. *The $\top, \bot$ semantics are defined as:*

- $s, i \models p$ *iff* $\rho^s_i = \top$ *or* $p \in \rho^s_i$.
- $s, i \models \neg\varphi$ *iff* $\overline{s}, i \not\models \varphi$.
- $s, i \models \varphi_1 \vee \varphi_2$ *iff* $s, i \models \varphi_1$ *or* $s, i \models \varphi_2$.
- $s, i \models \varphi_1 \mathcal{U}_I \varphi_2$ *iff there is* $j > i$ *with* $\tau^s_j - \tau^s_i \in I$ *such that* $s, j \models \varphi_2$ *and for every* k, $i < k < j$ $s, k \models \varphi_1$.
- $s, i \models \varphi_1 \mathcal{S}_I \varphi_2$ *iff either* $\rho^s_i = \top$ *or there is* j, $1 \leq j < i$ *with* $\tau^s_i - \tau^s_j \in I$ *such that* $s, j \models \varphi_2$ *and for all* k, $j < k < i$ $s, k \models \varphi_1$.

With these definitions it is possible to construct an infinite timed word $\top^\omega_{\varphi,s}$ for every formula φ and finite timed word s such that $s \cdot \top^\omega_{\varphi,s}, i \models \varphi$ iff $s, i \models^- \varphi$ and $s \cdot \overline{\top^\omega_{\varphi,s}}, i \models \varphi$ iff $s, i \models^+ \varphi$, and this is the main result of the section (Lemma 4). A complication that has to be dealt with is that the point-in-time semantics requires that a time is provided with each event. The remainder of this section is concerned with defining $\top^\omega_{\varphi,s}$ and establishing the correspondence with the weak and strong semantics given 1.

Definition 3. *Given a formula φ and timed word s, denote by $\top^\omega_{\varphi,s}$ the (infinite) timed word defined in the following way:*

- *Let $\mathcal{I}$ be the set of intervals I of the form (a,b) with $b \neq \infty$, occurring in subformulas of φ.*
- *Let $B = \bigcap_{\substack{1\leq i\leq |s|, I\in\mathcal{I} \\ \tau^s_{|s|}\in I+\tau^s_i}} I+\tau^s_i$. B is a finite intersection of open intervals and thus itself an open interval, and clearly $\tau^s_{|s|} \in B$, hence B is non-empty. Let t_0 be some rational number in B. Such t_0 exists because the rationals are dense in the real numbers.*
- *Let $d = \frac{d'}{2}$ where d' is the smallest length of an interval $I \in \mathcal{I}$, where the length of an interval (a,b) is given by $b-a$. Otherwise let $d = 1$ in case $\mathcal{I}$ is empty.*
- *Let T be the smallest set of times containing every time equivalent to one of:*
  - *$\left(t_0 - \tau^s_{|s|}\right) + \sum_l b_l \cdot n_l$ where $n_l \geq 0$ are integers and $b_1, b_2, \ldots$ are b such that there occurs an interval I of the form $[b,b]$ in some subformula of φ.*
  - *$d \cdot n_0 + \sum_{l\geq 1} b_l \cdot n_l$ where $n_l \geq 0$ are integers and $b_1, b_2, \ldots$ are such that for every i an interval $[b_i, b_i]$ occurs in some subformula of φ.*
  - *$\left(\tau^s_j - \tau^s_{|s|}\right) + \sum_l b_l \cdot n_l$ where j is an event in s with $\tau^s_{|s|} - \tau^s_j < b$ for b such that an interval $[b,b]$ occurs in some subformula of φ, $n_l \geq 0$ are integers and $b_1, b_2, \ldots$ are such that for every i an interval $[b_i, b_i]$ occurs in some subformula of φ.*
- *Finally, let $\top^\omega_{\varphi,s}$ be the timed word where $\tau^{\top^\omega_{\varphi,s}}$ is the sequence of times in T and $\rho_i^{\top^\omega_{\varphi,s}} = \top$ for every i.*

The above definition ensures that $\top^\omega_{\varphi,s}$ contains events in all relevant time intervals for the formula φ when the former is concatenated to the end of the time word s. This is made precise by Lemma 1.

$s^\top$ will be used to denote $s \cdot \top^\omega_{\varphi,s}$ when φ is clear from the context. Similarly $s^\perp$ will be used to denote $s \cdot \overline{\top^\omega_{\varphi,s}}$.

For example, if $\varphi = \Box_{(0,\infty)}\left(p\mathcal{U}_{(1,2)}q\right)$ and $\tau^s_{|s|-3} = 5, \tau^s_{|s|-2} = 9$, $\tau^s_{|s|-1} = 9.4$ and $\tau^s_{|s|} = 10$, then $\mathcal{I} = \{(1,2)\}$ and $\tau^s_{|s|-2} \in \tau^s_{|s|} + (1,2)$ and $\tau^s_{|s|-1} \in \tau^s_{|s|} + (1,2)$ so that we can pick $t_0 = 10.5$ and $d = \frac{d'}{2} = \frac{1}{2}$. $\top^\omega_{\varphi,s} = (\rho^{\top^\omega_{\varphi,s}}, \tau^{\top^\omega_{\varphi,s}})$ where $\rho^{\top^\omega_{\varphi,s}} = \top$ and $\tau_i^{\top^\omega_{\varphi,s}} = (10.5 - 10) + \frac{1}{2}\cdot(i-1)$ for every $i \geq 1$.

Lemma 1. *For every formula φ, subformula $\psi = \psi_1 \mathcal{U}_I \psi_2$ of φ, timed word s and integer $i \geq 0$ it holds that if either $i > |s|$ or $\tau_{|s|}^{s^\top} - \tau_i^{s^\top} < sup(I)$, then there is an event $j > |s|$ $(j > i)$ with $\tau_j^{s^\top} - \tau_i^{s^\top} \in I$.*

Proof. Consider the different forms of intervals I:

- $I = (a, b)$. First assume $i \leq |s|$ and $\tau_{|s|}^s \in (a, b) + \tau_i^s$, then by Definition 3 there is $j' \geq 1$ with $\tau_{j'}^{\top^\omega_{\varphi,s}} = t_0 - \tau_{|s|}^s$ so that $\tau_{|s|+j'}^{s^\top} = t_0$ and $t_0 \in (a, b) + \tau_i^s$, implying $\tau_{|s|+j'}^{s^\top} - \tau_i^{s^\top} \in (a, b)$. Otherwise we either have $i > |s|$, or we have $i \leq |s|$ and $\tau_{|s|}^s \notin I + \tau_i^s$. In the latter case we must have $\tau_{|s|}^s - \tau_i^s \leq a$ (because otherwise $\tau_{|s|}^s - \tau_i^s \geq b = sup(I)$, contradicting the premise). In either case, by Definition 3 there is $j' \geq 1$ with $\tau_{j'}^{\top^\omega_{\varphi,s}} = d \cdot n_0$ where $n_0 \geq 1$ is the smallest integer such that $\tau_i^{s^\top} + a < \tau_{|s|}^{s^\top} + d \cdot n_0$. Because $d < b - a$ we also have $\tau_{|s|}^{s^\top} + d \cdot n_0 < \tau_i^{s^\top} + b$, together implying $\tau_{|s|+j'}^{s^\top} - \tau_i^{s^\top} \in (a, b)$.
- $I = (a, \infty)$. By Definition 3 there is some $j' \geq 1$ with $\tau^{\top^\omega_{\varphi,s}} = d \cdot n_0$ where $n_0 \geq 0$ is integer such that $\tau_i^{s^\top} + a < \tau_{|s|}^{s^\top} + d \cdot n_0$, implying $\tau_{|s|+j'}^{s^\top} - \tau_i^{s^\top} \in I$.
- $I = [b, b]$. If $i \leq |s|$, then by the premise $\tau_{|s|}^s - \tau_i^s < b$ and by Definition 3 there is $j' \geq 1$ with $\tau_{j'}^{\top^\omega_{\varphi,s}} = \left(\tau_{|s|}^s - \tau_i^s\right) + \sum_l b_l \cdot n_l$ where $n_l = 1$ if $b_l = b$ and $n_l = 0$ otherwise, so that $\tau_{|s|+j'}^{s^\top} - \tau_i^{s^\top} = b$. Otherwise $i > |s|$ and by Definition 3 $\tau_i^{s^\top} = \tau_{|s|}^s + x + \sum_l b_n \cdot n_l$ where $n_l \geq 0$ are integers, but then there is $j > i$ having $\tau_j^{s^\top} = \tau_{|s|}^s + x + \sum_l b_l \cdot n'_l$ with $n'_l = n_l + 1$ for l if $b_l = b$ and $n'_l = n_l$ otherwise, so that $\tau_j^{s^\top} - \tau_i^{s^\top} = b$.

□

Lemma 2. *For every formula φ, subformula ψ of φ, and timed word s and $i \geq 1$, the following are true:*

- $s^\top, |s| + i \models \psi$, *and*
- $\overline{s^\top}, |s| + i \not\models \psi$.

Proof. Induction on ψ.

- p. The premise has $|s| + i > |s|$ and $\rho_{|s|+i}^{s^\top} = \rho_i^{\top^\omega_{\varphi,s}} = \top$, $\rho_{|s|+i}^{\overline{s^\top_{\varphi,s}}} = \rho_i^{\overline{\top^\omega_{\varphi,s}}} = \bot$.
- $\neg\psi_1$. By induction hypothesis $s^\top, |s| + i \models \psi_1$, $\overline{s^\top}, |s| + i \not\models \psi_1$.

- $\psi_1 \vee \psi_2$. By induction hypothesis: $s^\top, |s|+i \models \psi_1$ and $s^\top, |s|+i \models \psi_2$, $\overline{s^\top}, |s|+i \not\models \psi_1$ and $\overline{s^\top}, |s|+i \not\models \psi_2$.

- $\psi_1 \mathcal{U}_I \psi_2$.

  - By Lemma 1 there is $j > |s|+i$ such that $\tau_j^{s^\top} - \tau_i^{s^\top} \in I$ and by induction hypothesis $s^\top, j \models \psi_2$ and for every k, $i < k < j$, $s^\top, k \models \psi_1$, so that $s^\top, i \models \psi_1 \mathcal{U}_I \psi_2$.
  - By induction hypothesis every $j > |s|+i$ has $\overline{s^\top}, j \not\models \psi_2$, implying $\overline{s^\top}, |s|+i \not\models \psi_1 \mathcal{U}_I \psi_2$.

- $\psi_1 \mathcal{S}_I \psi_2$. Every $|s|+i > |s|$ so $\rho^{s^\top}_{|s|+i} = \rho_i^{\top^\omega_{\varphi,s}} = \top$, $\rho^{\overline{s^\top}}_{|s|+i} = \rho_i^{\overline{\top^\omega_{\varphi,s}}} = \bot$.

□

Lemma 3. *For every formula φ and subformula of φ of the form $\varphi_1 \mathcal{U}_I \varphi_2$ we have $s^\top, i \models \varphi_1 \mathcal{U}_I \varphi_2$ iff one of the following holds:*

- *There is j, $i < j \leq |s|$ with $\tau_j^{s^\top} - \tau_i^{s^\top} \in I$ such that $s^\top, j \models \varphi_2$, and for every k, $i < k < j$ $s^\top, k \models \varphi_1$.*

- *$i > |s|$.*

- *$\tau^{s^\top}_{|s|} - \tau_i^{s^\top} < sup(I)$ and for all k, $i < k \leq |s|$ $s^\top, k \models \varphi_1$.*

Proof. $\Longrightarrow$ Assume $s^\top, i \models \varphi_1 \mathcal{U}_I \varphi_2$. If $i > |s|$ the proof is done, otherwise the proof proceeds. By definition there is $j > i$, $\tau_j^{s^\top} - \tau_i^{s^\top} \in I$ such that $s^\top, j \models \varphi_2$ and for every k, $i < k < j$ $s^\top, k \models \varphi_1$. If $j \leq |s|$, then for every k, $i < k < j$, $s^\top, k \models \varphi_1$. Otherwise $j > |s|$, $\tau^{s^\top}_{|s|} - \tau_i^{s^\top} < \tau_j^{s^\top} - \tau_i^{s^\top} < sup(I)$ and for every k, $i < k \leq \tau^{s^\top}_{|s|}$ $s^\top, k \models \varphi_1$.

$\Longleftarrow$ Three cases:

- If there is j, $i < j \leq |s|$ with $\tau_j^{s^\top} - \tau_i^{s^\top} \in I$ such that $s^\top, j \models \varphi_2$ and for every k, $i < k < j$ $s^\top, k \models \varphi_2$, then it follows that $s^\top, i \models \varphi_1 \mathcal{U}_I \varphi_2$.

- If $i > |s|$ then $s^\top, i \models \varphi_1 \mathcal{U}_I \varphi_2$ trivially by Lemma 2.

- If $\tau^{s^\top}_{|s|} - \tau_i^{s^\top} < sup(I)$ and for every k, $i < k \leq |s|$ $s^\top, k \models \varphi_1$, then by Lemma 1 there is $j > |s|$ with $\tau_j^{s^\top} - \tau_i^{s^\top} \in I$ and by Lemma 2 $s^\top, j \models \varphi_2$, and by Lemma 2 every k', $k < k' < j$ $s^\top, k' \models \varphi_1$, so that $s^\top, i \models \varphi_1 \mathcal{U}_I \varphi_2$.

□

Finally this is the main result of this subsection:

Lemma 4. *For every subformula ψ of φ and every timed word s it holds that:*

- $s, i \models^- \psi$ *iff* $s^\top, i \models \psi$.
- $s, i \models^+ \psi$ *iff* $s^\perp, i \models \psi$.

Proof.

- p.
  - $s, i \models^- p$
    $\iff i > |s|$ or $p \in \rho_i^s$
    $\iff$ Either $i > |s|$ and $\rho_i^{s^\top} = \rho_{i-|s|}^{\top^\omega_{\varphi,s}} = \top$, or $i \leq |s|$ and $p \in \rho_i^{s^\top} = \rho_i^s$
    $\iff s^\top \models p$
  - $s, i \models^+ p \iff i \leq |s|$ and $p \in \rho_i^s$
    $\iff p \in \rho_i^{s^\perp}$
    $\iff s^\perp \models p$
- $\neg\varphi$
  - $s, i \models^- \neg\varphi$
    $\iff s, i \not\models^+ \varphi$
    $\iff$ [Induction]
    $\iff \overline{s^\top}, i \not\models \varphi$
    $\iff s^\top, i \models \neg\varphi$
  - $s, i \models^+ \neg\varphi$
    $\iff s, i \not\models^- \varphi$
    $\iff$ [Induction]
    $\iff \overline{s^\perp}, i \not\models \varphi$
    $\iff s^\perp, i \models \neg\varphi$
- $\varphi_1 \vee \varphi_2$.
  - $s, i \models^- \varphi_1 \vee \varphi_2$
    $\iff s, i \models^- \varphi_1$ or $s, i \models^- \varphi_2$
    $\iff$ [Induction]
    $\iff s^\top, i \models \varphi_1$ or $s^\top, i \models \varphi_2$
    $\iff s^\top, i \models \varphi_1 \vee \varphi_2$

–

$s, i \models^+ \varphi_1 \vee \varphi_2$
$\iff s, i \models^+ \varphi_1$ or $s, i \models^+ \varphi_2$
$\iff$ [Induction]
$\iff s^\perp, i \models \varphi_1$ or $s^\perp, i \models \varphi_2$
$\iff s^\perp, i \models \varphi_1 \vee \varphi_2$

- $\varphi_1 \mathcal{U}_I \varphi_2$

  –

  $s, i \models^- \varphi_1 \mathcal{U}_I \varphi_2$
  $\iff$ [By applying the induction hypothesis to Lemma 3]
  $\iff s\top, i \models \varphi_1 \mathcal{U}_I \varphi_2$

  –

  $s, i \models^+ \varphi_1 \mathcal{U}_I \varphi_2$
  $\iff$ There is j, $i < j \leq |s|$ with $\tau_j^{s^\perp} - \tau_i^{s^\perp} \in I$ such that $s, j \models^+ \varphi_2$ and for every k, $i < k < j$ $s, k \models^+ \varphi_1$
  $\iff$ [Induction]
  $\iff$ There is $j > i$ with $\tau_j^{s^\perp} - \tau_i^{s^\perp} \in I$ such that $s^\top, j \models \varphi_2$ and for every k, $i < k < j$ $s^\top, k \models \varphi_1$
  $\iff s^\perp, i \models \varphi_1 \mathcal{U}_I \varphi_2$

- $\varphi_1 \mathcal{S}_I \varphi_2$

  –

  $s, i \models^- \varphi_1 \mathcal{S}_I \varphi_2$
  $\iff$ Either $i > |s|$ or there is j, $1 \leq j < i$ with $\tau_i^s - \tau_j^s \in I$ such that $s, j \models^- \varphi_2$ and for every k, $j < k < i$ $s, k \models^- \varphi_1$
  $\iff$ [Induction]
  $\iff$ Either $\rho_i^{s^\top} = \top$ or there is $1 \leq j < i$, $\tau_i^{s^\top} - \tau_j^{s^\top} \in I$ such that $s^\top, j \models \varphi_2$ and for every k, $j < k < i$, $s^\top, k \models \varphi_1$
  $\iff s^\top, i \models \varphi_1 \mathcal{S}_I \varphi_2$

–

$s, i \models^+ \varphi_1 \mathcal{S}_I \varphi_2$

$\iff i < |s|$ and there is j, $1 \leq j < i$ with $\tau_i^s - \tau_j^s \in I$ such that $s, j \models^+ \varphi_2$ and for every k, $j < k < i$, $s, k \models^+ \varphi_1$.

$\iff$ [Induction]

$\iff$ There is $j < i$ with $\tau_i^{s^\perp} - \tau_j^{s^\perp} \in I$ such that $s^\perp, j \models \varphi_2$ and for every k, $j < k < i$ $s^\perp, k \models \varphi_1$

$\iff s^\perp, i \models \varphi_1 \mathcal{S}_I \varphi_2$

□

Lemma 4 establishes equivalence between the weak and strong semantics of Section 2.1 and the $\top, \perp$ semantics. We thus have a single relation expressing our target semantics, and the objective of this chapter has been reached.

Chapter 3

Timed automata

Before continuing to the translation of MTL to timed automata and finally a monitoring procedure, which is the topic of the next chapter, it is necessary to give some background on the timed automata used.

This chapter gives some background information to the timed automata used in the construction presented in Section 4. The timed automata used here differ somewhat from those introduced in [1]. The exposition given here is meant to provide a working understanding of their operation and consists of a condensed presentation of the definitions given in [25] to which the interested reader is referred for a complete formal definition.

3.1 Time, signals, clocks and constraints

Times t are represented by the non-negative reals. A signal $w : \mathbb{R}_{\geq 0} \to \Sigma$ maps each time t to a letter in the input alphabet Σ.

The set of valuations of a set $\mathcal{C} = \{x_1, \ldots, x_n\}$ of clock variables, each denoted as $v = (v_1, \ldots, v_n)$ defines the *clock space* $\mathcal{H} = (\mathbb{R}^+ \cup \{\perp\})$. ($\perp$ is used here to represent the value of an inactive clock). The set $\mathcal{A}(\mathcal{A})$ of *atomic clock constraints* over $\mathcal{C}$ consists of conditions of the form $x \bowtie y+d$ where x and y are clock variables, $\bowtie \in \{<, >, \leq, \geq\}$ and d is an integer. The set of positive Boolean formulas over X, $\mathbb{B}^+(X)$ is the set of Boolean formulas built from elements of X using $\wedge$ and $\vee$. Finally $\mathbb{C}(\mathcal{C}) = \mathbb{B}^+(\mathcal{A}(\mathcal{C}))$ is the set of *clock constraints* over $\mathcal{C}$.

3.2 Timed automata

A timed automaton (TA) is $\mathcal{A} = (\Sigma, Q, \mathcal{C}, \lambda, I, \Delta, q_0, \mathcal{F})$, with input alphabet Σ, set of states Q, set of clocks variables $\mathcal{C}$, input labeling function $\lambda : Q \cup \Delta \to \Sigma$ associating every state and transition with an input symbol, staying condition

function $I : Q \to \mathbb{C}(\mathcal{C})$ associating every state with a clock constraint, transition relation Δ, initial state $q_0 \in Q$, sets of accepting states $\mathcal{F} \subseteq 2^Q$. The transition relation Δ consists of tuples of the form (q, g, ρ, q') where $q, q' \in Q$, $g \in \mathbb{C}(\mathcal{C})$ is a clock constraint and ρ is a clock update function defined by an assignment of the form $x := 0$, $x := \perp$ or $x := y$ or a set of such assignments.

Note that our definition of λ differs from [25] in that the domain of λ is $Q \cup \Delta$ (instead of only Q) so that both states and transitions are labeled. This is done in order to handle *punctuality* in the signal w, by which is meant that a proposition p may hold precisely at time t ($p \in w[t]$) but neither directly before nor after t. Every interval containing t contains a time $t' \neq t$ with $p \notin w[t']$. It is required that a transition $\delta \in \Delta$ may be taken at time t only if $\lambda(\delta) \in w^s[t]$. This addition results in a slightly altered definition of what it means for a TA to be deterministic compared to in [25]:

Definition 4. *A deterministic timed automaton is an automaton whose guards and staying conditions satisfy:*

1. *For every two distinct transitions $\delta_1 = (q, g_1, \rho_1, q_1)$ and $\delta_2 = (q, g_2, \rho_2, q_2)$ either $\lambda(q_1) \neq \lambda(q_2)$ or $\lambda(\delta_1) \neq \lambda(\delta_2)$ or $g_1 \wedge g_2$ is unsatisfiable.*

2. *For every transition $\delta = (q, g, \rho, q')$, either $\lambda(q) \neq \lambda(q')$ or $\lambda(q) \neq \lambda(\delta)$ or the intersection of g and $I(q)$ is either empty or isolated, i.e. there does not exist an open interval (t, t') such that $(t, t') \subseteq I(q)$ and $(t, t') \cap g \neq \emptyset$.*

A run of $\mathcal{A}$ begins in state q_0 and initially all clocks are inactive. The run consists of a strict alternation between two kinds of steps while $\mathcal{A}$ reads the input signal w:

1. Time progress steps which are open time intervals during which the run remains in a state q and the input labeling function $\lambda(q)$ and staying condition $I(q)$ are satisfied continuously.

2. Discrete instantaneous transitions where the run takes a transition $\delta = (q, g, \rho, q') \in \Delta$ such that the guard g is satisfied at the time of transition and the clock variables are updated using the clock update function ρ.

Using generalized Büchi acceptance condition the run is accepting if the set of infinitely often visited states contains at least one state from every accepting set $\mathcal{F}_i \in \mathcal{F}$.

3.2.1 Example TA

Consider the TA $\mathcal{A}_{EX}$ depicted in Figure 3.1. Formally $\mathcal{A}_{EX} = (\Sigma, Q, \mathcal{C}, \lambda, I, \Delta, q_0, \mathcal{F})$ where $\Sigma = \{a, b, c\}$, $\mathcal{C} = \{x, y\}$, $\lambda(q_1) = a$, $\lambda(q_2) = b$, $I(q_1) = x > 0 \wedge x < 1$, $I(q_2) = y > 0 \wedge y < 1$, $\mathcal{F} = \{Q\}$ and Δ consists of the following transitions:

- $\delta_1 = (q_0, true, \{x := 0\}, q_1)$, $\lambda(\delta_1) = c$.
- $\delta_2 = (q_0, true, \{y := 0\}, q_2)$, $\lambda(\delta_2) = c$.
- $\delta_3 = (q_1, x = 1, \{y := 0\}, q_2)$, $\lambda(\delta_3) = c$.
- $\delta_4 = (q_1, x = 1, \{x := 0\}, q_1)$, $\lambda(\delta_4) = c$.
- $\delta_5 = (q_2, y = 1, \{x := 0\}, q_1)$, $\lambda(\delta_5) = c$.
- $\delta_6 = (q_2, y = 1, \{y := 0\}, q_1)$, $\lambda(\delta_6) = c$.

$\mathcal{A}_{EX}$ accepts w iff $w[t] = c$ for every integral t, and $w[t'] = a$ for every $t' \in (t, t+1)$ or $w[t'] = b$ for every $t' \in (t, t+1)$. To see this, first note that clock x is associated wit state q_1 and clock y is associated with state q_2 in the sense that:

- The staying condition of q_1 is $0 < x < 1$, every incoming transition to q_1 assigns $x := 0$ and every outgoing transition from q_1 has guard $x = 1$.
- The staying condition of q_2 is $0 < y < 1$, every incoming transition to q_2 assigns $y := 0$ and every outgoing transition from q_2 has guard $y = 1$.

Thus if a run of $\mathcal{A}_{EX}$ takes an incoming transition to q_1 at time t it must remain in q_1 until time $t + 1$ when $x = 1$ and there are eligible outgoing transitions from q_1. In addition for every $t' \in (t, t + 1)$ the label $\lambda(q_1) = a$ requires that $w[t'] = a$ or the run terminates. Conversely if a run of $\mathcal{A}_{EX}$ takes an incoming transition to q_2 at time t it must remain in q_2 until time $t + 1$ when $y = 1$ and there are eligible outgoing transitions from q_1. In addition for every $t' \in (t, t + 1)$ the label $\lambda(q_2) = b$ requires that $w[t'] = b$ or the run terminates.

The initial transitions to either of q_1 and q_2 occur at $t = 0$. It follows that $\mathcal{A}_{EX}$ accepts w only if $w[t] = c$ whenever t is integral.

3.3 Dependent Timed Automata

A dependent timed automaton is a transducer of runs of TAs. The DTA has no clocks of its own but instead depends on passively reading clocks of other TAs.

More formally a dependent timed automaton (DTA) is $D = (\Sigma, \Gamma, Q, \mathcal{C}, \gamma, I, \Delta, q_0, \mathcal{F})$ where Σ, Q, $\mathcal{C}$, q_0 and $\mathcal{F}$ are as in timed automata, Γ is the output alphabet, the output labeling function $\gamma : Q \to \Gamma$ associates each state with an output letter, and the staying condition function $I : Q \to \mathbb{B}^+(\mathbb{A}(\mathcal{C}) \cup \Sigma)$ associates every state with a Boolean combination of atomic clock constraints and input letters. The transition relation Δ consists of tuples (q, g, o, q') where $q, q' \in Q$, $g \in \mathbb{B}^+(\mathbb{A}(\mathcal{C}) \cup \Sigma)$ is a

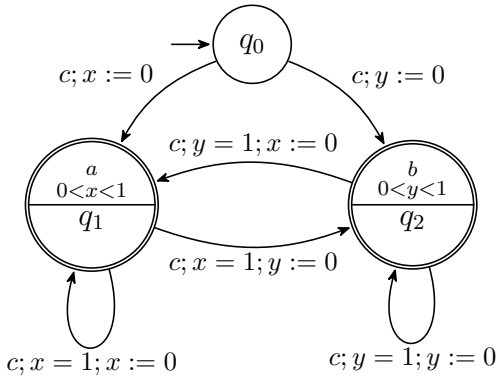

Figure 3.1: $\mathcal{A}_{EX}$ accepts all w such that $w[t] = c$ iff t is integral and for every integral t either $w[t'] = a$ for every $t' \in (t, t+1)$, or $w[t'] = b$ for every $t' \in (t, t+1)$.

Boolean combination of input letters and atomic clock constraints and $o \in \Gamma$ is an output letter associated with the transition.

A run of D begins in state q_0. The run consists of a strict alternation between two kinds of steps while D reads the input signal w and clocks $\mathcal{C}$:

1. Time progress steps which are open time intervals during which the run remains in a state q and the staying condition I_q is satisfied continuously.

2. Discrete instantaneous transitions where the run takes a transition $\delta = (q, g, o, q') \in \Delta$ such that the guard g is satisfied at the time of transition.

3.3.1 Example DTA

Consider the DTA D_{EX} depicted in Figure 3.2. Formally $D_{EX} = (\Sigma, \Gamma, Q, \mathcal{C}, \gamma, I, \Delta, q'_0, \mathcal{F})$ where $\Sigma = \{a, b, c\}$, $\Gamma = \{o\}$, $Q = \{q'_0, q'_1, q'_2\}$, $\mathcal{C} = \{x, y\}$, $\gamma(q'_1) = \gamma(q'_2) = o$, $I(q'_1) = (x > 0) \wedge (x < 1)$, $I(q'_2) = (y > 0) \wedge (y < 1)$, $\mathcal{F} = \{Q\}$ and Δ consists of the following transitions:

- $\delta_1 = (q'_0, x = 0, o, q'_1)$.
- $\delta_2 = (q'_0, y = 0, o, q'_2)$.
- $\delta_3 = (q'_1, y = 0, o, q'_2)$.
- $\delta_4 = (q'_2, x = 0, o, q'_1)$.

Looking at D_{EX} we see that clock x is associated with state q'_1 and clock y is associated with state q'_2 in the sense that:

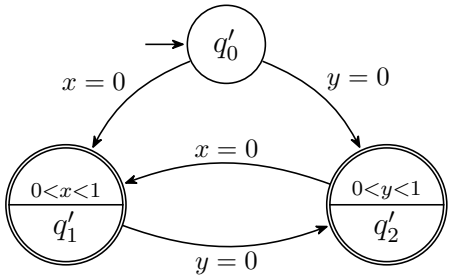

Figure 3.2: Example DTA D_{EX}. The state sequence of every accepting run of D_{EX} start with q'_0 followed by infinite repetition of strict alternation between q'_1 and q'_2, that is either of the infinite sequences $q'_0 q'_1 q'_2 \ldots$ and $q'_0 q'_2 q'_1 \ldots$

- The staying condition of q'_1 is $0 < x < 1$ and incoming transitions to q'_1 have guard $x = 0$.
- The staying condition of q'_2 is $0 < y < 1$ and incoming transitions to q'_2 have guard $y = 0$.

Because the initial state q'_0 has transitions to both of q'_1 and q'_2 and the only outgoing transition from q'_1 is to q'_2 and the only outgoing transition from q'_2 is to q'_1 it follows that the state sequences of every accepting run of D_{EX} starts with q'_0 followed by infinite repetition of strict alternation of q'_1 and q'_2.

3.4 Compositions

Composition operators allows constructing TAs by combining several smaller DTAs and these are used extensively in the construction in the next section. Here we introduce the two kinds of composition that will be used later, namely *parallel* and *sequential* composition.

Given two TAs $\mathcal{A}_1$ and $\mathcal{A}_2$ whose sets of clock variables and input alphabets are disjoint, their parallel composition $\mathcal{A}_3 = \mathcal{A}_1 \| \mathcal{A}_2$ is another TA that makes the same clock assignments as $\mathcal{A}_1$ and $\mathcal{A}_2$. Intuitively $\mathcal{A}_3$ can be understood as internally evaluating $\mathcal{A}_1$ and $\mathcal{A}_2$, each of which reads its section of the input signal while making clock assignments and taking transitions synchronously or asynchronously to one another.

The sequential composition operator $\otimes$ gives a way to combine DTAs with DTAs and TAs with DTAs. If D_1 and D_2 are DTAs, the composition $D_1 \otimes D_2$ is another DTA which passes the input signal to D_1, then extends the input given to D_1 with the output from D_1 and passes it as input to D_2. Similarly if $\mathcal{A}_1$ is a TA and D_2 is a DTA, the composition $\mathcal{A}_1 \otimes D_2$ is a TA which passes the input to $\mathcal{A}_1$, then passes the same input together with the clocks of $\mathcal{A}_1$ into D_2.

3.4.1 Example: The composition $\mathcal{A}_{EX} \otimes D_{EX}$

Because $\mathcal{A}_{EX}$ and D_{EX} have identical input alphabets Σ and set of clocks $\mathcal{C}$ the composition $\mathcal{A}_{EX} \otimes D_{EX}$ is defined and is a TA with input alphabet Σ and set of clocks $\mathcal{C}$. This TA accepts w iff $\mathcal{A}_{EX}$ accepts w and D_{EX} accepts w and the clock assignments made by $\mathcal{A}_{EX}$. D_{EX} can be thought of as acting as a filter on the set of accepting runs of $\mathcal{A}_{EX}$.

Every infinite run of $\mathcal{A}_{EX} \otimes D_{EX}$ is accepting. Consider such a run and let $\xi^{\mathcal{A}}$ and ξ^{D} be the corresponding runs of $\mathcal{A}_{EX}$ and D_{EX} respectively. At any time t it is true that:

- the state of $\xi^{\mathcal{A}}$ is q_1 iff the state of ξ^{D} is q_1', and
- the state of $\xi^{\mathcal{A}}$ is q_2 iff the state of ξ^{D} is q_2'.

This follows because initially $t = 0$ ξ^{D} transitions to q_1' iff $x = 0$ and $x = 0$ iff $\xi^{\mathcal{A}}$ transitions to q_1, and later $\xi^{\mathcal{A}}$ transitions to q_2 at the same time as ξ^{D} transitions to q_2' when $x = 1$. Had $\xi^{\mathcal{A}}$ instead taken a transition to q_1 when $x = 1$ it would assign $x := 1$ but this contradicts the staying condition of q_1'. The argument for q_2 and q_2' is symmetrical.

It follows that $\mathcal{A}_{EX} \otimes D_{EX}$ accepts w iff $w[t] = c$ for every integral t and either of:

- $w[t'] = a$ for every $t' \in (t, t+1)$ if t is even and $w[t'] = b$ for every $t' \in (t, t+1)$ if t odd, or
- $w[t'] = b$ for every $t' \in (t, t+1)$ if t is even and $w[t'] = a$ for every $t' \in (t, t+1)$ if t odd.

3.5 Conclusion

In the previous chapter we obtained semantics that express our intent and can be practically implemented. Having gained working understanding of the timed automata that will be used in our construction and having seen how complex TAs can be built from simpler TAs by using compositions we are now ready to proceed to the construction itself, which is the topic of the next chapter.

Chapter 4

Translation from point-in-time MTL to timed automata

In the previous chapters we defined practicable semantics for our monitoring procedure and introduced timed automata. With these preliminaries out of the way we are ready to describe the main result of this thesis, which is a construction based on timed automata which implements the target semantics given in Section 2.2.

The main idea underpinning the construction is that TAs can memorize part of an input word s packaged as a signal w^s (Section 4.1), and that by memorizing every event and proposition of w^s in a sliding time window (Section 4.3), the length of which depends on the structure of φ (Section 4.2), it is possible to construct predicates, all of which are valid clock constraints, for subformulas ψ of φ where ψ is either an atomic proposition, a negation, a disjunction or a bounded temporal operator (Section 4.4). If ψ is an unbounded temporal operators the situation is more complicated, but by associating a specially constructed DTA with ψ a predicate can be constructed for ψ as well (Section 4.5). Using these building blocks it is possible to construct a TA $\mathcal{A}_\varphi$ which accepts w^s iff $s, 1 \models \varphi$. Correctness and complexity for the construction is given in Section 4.7 and a monitoring procedure based on the construction is outlined in Section 4.8.

The construction from [25] is used as a starting point. Throughout the chapter φ is used to denote the MTL formula being translated and ψ is used to refer to specific subformulas of φ.

4.1 Packaging a timed word as a signal

Our semantics are based on timed words, but TAs read signals. It is thus necessary to provide a translation from timed word to signal.

Definition 5. *Given a timed word $s = (\rho^s, \tau^s)$ and formula φ, let $T_i = [\tau_i^s, \tau_{i+1}^s)$ if $i < |s|$ and $T_i = [\tau_{|s|}^s, \infty)$ if $i = |s|$. The signal w^s is constructed in the following way:*

- $\top \in w^s[t]$ *iff* $\rho_i^s = \top$ *and* $t \in T_i$ *for some* i.
- $\bot \in w^s[t]$ *iff* $\rho_i^s = \bot$ *and* $t \in T_i$ *for some* i.
- $p \in w^s[t]$ *iff* $p \in \rho_i^s$ *and* $t \in T_i$ *for some* i.
- $\tau \in w^s[t]$ *iff* $t = \tau_i^s$ *for some* i.

It should be clear from Definition 5 that $\top \in w^s[t]$, $\bot \in w^s[t]$ and $p \in w^s[t]$ are pairwise mutually exclusive. We say that an event occurs at time t if $\tau \in w^s[t]$.

4.2 Past and future reach

Evaluating an MTL formula φ at an event i in general requires evaluating subformulas ψ of φ at events relative to i. For example if $\varphi = \psi_1 \mathcal{U}_I \psi_2$ evaluating $s, i \models \varphi$ in general requires evaluating ψ_1 and ψ_2 at every event j with $\tau_j^s - \tau_i^s \in I$. (See Figure 4.1 and Figure 4.2 for the case $I = (a, b)$). If either of ψ_1 or ψ_2 have subformulas of their own that are also temporal operators the subformulas of these subformulas will need to be evaluated at events possibly further into the future or past relative i. The past and future horizon of φ, $pas(\varphi)$ and $fut(\varphi)$ gives a maximum number of time units in the past and future that atomic propositions need to be known in order to evaluate φ.

Definition 6. *The past reach and future reach of a subformula ψ, denoted by $pas(\psi)$ and $fut(\psi)$, are given by:*

- $pas(p) = 0$.
- $pas(\neg\psi) = pas(\psi)$.
- $pas(\psi_1 \vee \psi_2) = max(pas(\psi_1), pas(\psi_2))$.
- $pas(\psi_1 \mathcal{U}_I \psi_2) = max\left(pas(\psi_1), pas(\psi_2) - a\right)$ *for* $I = (a, b)$, $I = (a, \infty)$ *or* $I = [a, a]$.
- $pas(\psi_1 \mathcal{S}_I \psi_2) = b + max(pas(\psi_1), pas(\psi_2))$ *for* $I = (a, b)$, $I = [b, b]$.
- $pas(\psi_1 \mathcal{S}_I \psi_2) = a + 3 + max(pas(\psi_1), pas(\psi_2))$ *for* $I = (a, \infty)$.
- $fut(p) = 0$.

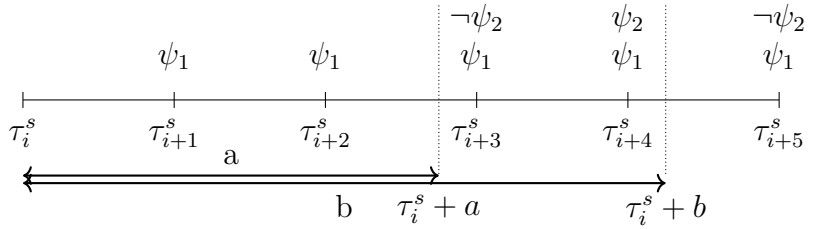

Figure 4.1: Evaluating $s, i \models \psi_1 \mathcal{U}_{(a,b)} \psi_2$ in general requires evaluating $s, k \models \psi_1$ for every $k > i$ such that $\tau_k^s - \tau_i^s < b$ and every $s, j \models \psi_2$ such that $j > i$ and $\tau_j^s - \tau_i^s \in (a, b)$.

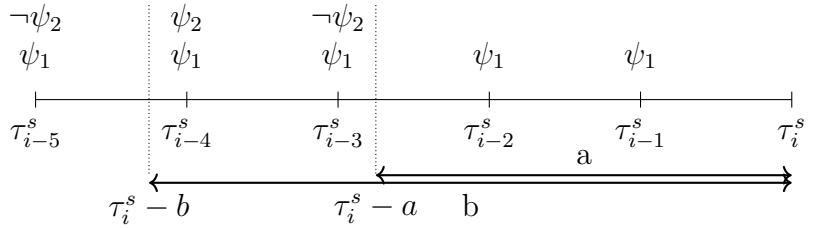

Figure 4.2: Evaluating $s, i \models \psi_1 \mathcal{S}_{(a,b)} \psi_2$ in general requires evaluating $s, k \models \psi_1$ for every $k < i$ such that $\tau_i^s - \tau_k^s < b$ and every $s, j \models \psi_2$ such that $j < i$ and $\tau_i^s - \tau_j^s \in (a, b)$.

- $fut(\neg\psi) = fut(\psi)$.
- $fut(\psi_1 \vee \psi_2) = max(fut(\psi_1), fut(\psi_2))$.
- $fut(\psi_1 \mathcal{U}_I \psi_2) = b + max(fut(\psi_1), fut(\psi_2))$ *for* $I = (a, b)$ *or* $I = [b, b]$.
- $fut(\psi_1 \mathcal{U}_I \psi_2) = a + 2 + max(fut(\psi_1), fut(\psi_2))$ *for* $I = (a, \infty)$.
- $fut(\psi_1 \mathcal{S}_I \psi_2) = max\left(fut(\psi_1), fut(\psi_2) - a\right)$ *for* $I = (a, b)$, $I = (a, \infty)$ *or* $I = [a, a]$.

For the remainder of this chapter let $h = pas(\varphi) + fut(\varphi)$. This quantity is the length of the sliding time window maintained by the construction. If ψ is a subformula we call $pas(\psi)$ the *past reach* of ψ and $fut(\psi)$ the *future reach* of ψ.

4.2.1 Example

We now consider a formula which uses a large part of our functionality:

$$\begin{aligned} \varphi &= \Box_{(0,\infty)} \left\{ \left(p \, \mathcal{S}_{(1,10)} q \right) \implies \left(r \, \mathcal{U}_{(5,10)} r \vee s \right) \right\} \\ &= \neg \left\{ true \, \mathcal{U}_{(0,\infty)} \neg \left(\left[p \, \mathcal{S}_{(1,10)} q \right] \vee \neg \left[r \, \mathcal{U}_{(5,10)} s \right] \right) \right\} \end{aligned} \tag{1}$$

By mechanically applying Definition 6 we obtain $pas(\varphi)$ and $fut(\varphi)$:

$$\begin{aligned}
pas(\varphi) =& pas\left(\neg\left\{true\,\mathcal{U}_{(0,\infty)}\neg\left(\left[p\,\mathcal{S}_{(1,10)}q\right]\vee\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right\}\right)\\
=& pas\left(true\,\mathcal{U}_{(0,\infty)}\neg\left(\left[p\,\mathcal{S}_{(1,10)}q\right]\vee\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right)\\
=& max\left(pas(true), pas\left(\neg\left(\left[p\,\mathcal{S}_{(1,10)}q\right]\vee\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right)\right)\\
=& max\left(0, pas\left(\left[p\,\mathcal{S}_{(1,10)}q\right]\vee\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right)\\
=& max\left(0, max\left(pas\left(p\,\mathcal{S}_{(1,10)}q\right), pas\left(\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right)\right)\\
=& max\left(0, max\left(pas\left(p\,\mathcal{S}_{(1,10)}q\right), pas\left(r\,\mathcal{U}_{(5,10)}s\right)\right)\right)\\
=& max\left(0, max\left(max\left(pas(p), pas(q)\right)+10, max\left(pas(r), pas(s)-5\right)\right)\right)\\
=& max\left(0, max\left(max\left(0,0\right)+10, max\left(0,-5\right)\right)\right)\\
=& max\left(0, max\left(10,0\right)\right)\\
=& max\left(0,10\right)\\
=& 10
\end{aligned}$$

$$\begin{aligned}
fut(\varphi) =& fut\left(\neg\left\{true\,\mathcal{U}_{(0,\infty)}\neg\left(\left[p\,\mathcal{S}_{(1,10)}q\right]\vee\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right\}\right)\\
=& fut\left(true\,\mathcal{U}_{(0,\infty)}\neg\left(\left[p\,\mathcal{S}_{(1,10)}q\right]\vee\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right)\\
=& max\left(fut(true), fut\left(\neg\left(\left[p\,\mathcal{S}_{(1,10)}q\right]\vee\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right)\right)+2\\
=& max\left(0, fut\left(\left[p\,\mathcal{S}_{(1,10)}q\right]\vee\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right)+2\\
=& max\left(0, max\left(fut\left(p\,\mathcal{S}_{(1,10)}q\right), fut\left(\neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)\right)\right)+2\\
=& max\left(0, max\left(fut\left(p\,\mathcal{S}_{(1,10)}q\right), fut\left(r\,\mathcal{U}_{(5,10)}s\right)\right)\right)+2\\
=& max\left(0, max\left(max\left(fut(p), fut(q)-1\right), max\left(fut(r), fut(s)\right)+10\right)\right)+2\\
=& max\left(0, max\left(max\left(0,-1\right), max\left(0,0\right)+10\right)\right)+2\\
=& max\left(0, max\left(0,10\right)\right)+2\\
=& max\left(0,10\right)+2\\
=& 12
\end{aligned}$$

Thus $h = pas(\varphi) + fut(\varphi) = 22$. See Figure 4.3 for an illustration of the time interval relative event i where atomic propositions need to be evaluated in order to evaluate $s, i \models \varphi$.

4.3 Memorization of atomic propositions and events

We saw in Section 4.2 that every formula φ has a past reach $pas(\varphi)$ and future reach $fut(\varphi)$ which is the number of time units in the past and future relative to some event i where other events may affect the truth of $s, i \models \varphi$. In this section we construct a TA which memorizes the time of every event and for every atomic

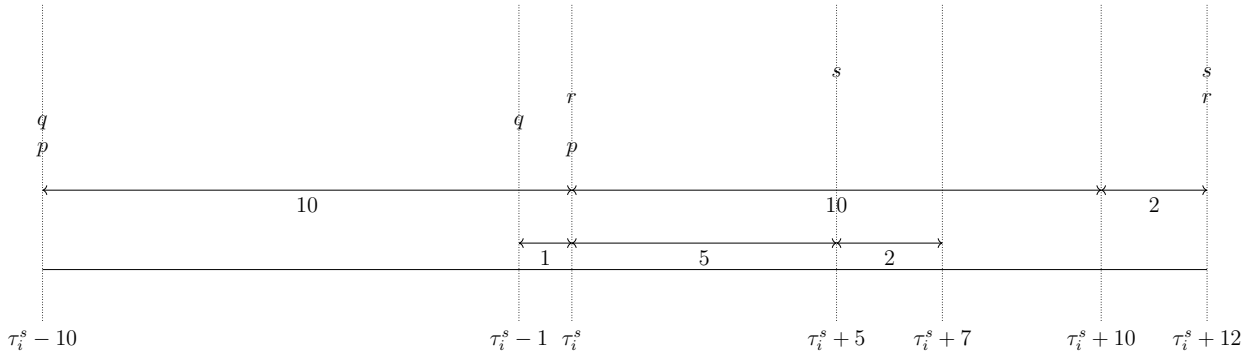

Figure 4.3: Past and future horizon of $\Box_{(0,\infty)}\left\{\left(p\,\mathcal{S}_{(1,10)}q\right) \implies \left(r\,\mathcal{U}_{(5,10)}r \vee s\right)\right\}$. To evaluate it we need to evaluate $s, j \models p$ for $j < i$ such that $\tau_i^s - \tau_j^s \in (0, 10)$, $s, j \models q$ for $j < i$ such that $\tau_i^s - \tau_j^s \in (1, 10)$, $s, j \models r$ for $j > i$ such that $\tau_j^s - \tau_j^s \in (0, 10 + 2)$ and $s, j \models s$ for $j > i$ such that $\tau_j^s - \tau_i^s \in (5, 10 + 2)$. The "+2" is due to the unbounded until.

proposition the start and end of every subinterval in which the proposition held in a sliding window of size $h = pas(\varphi) + pas(\varphi)$ relative to the current time.

4.3.1 TA for punctual events

In order to make memorization of punctual events possible using a finite number of clocks it is assumed there is a known maximum *event rate* k_{ER}. The sliding window in which events need to be memorized stretches from h time units before the current time t and up until t, namely the interval $[t - h, t]$. Because of the bounded event rate there are at most $l = \lceil k_{ER} \cdot h \rceil$ such events. The TA for punctual events is thus $\mathcal{A}_\tau = \left(2^{\{\tau\}}, Q, \mathcal{C}, \lambda, I, \Delta, q_0, \{Q\}\right)$ where $Q = \{q_0, q_1\}$, $\mathcal{C} = \{x_1^\tau, \ldots, x_l^\tau\}$, $I(q_1) = true$, $\lambda(q_1) = \neg\tau$ where Δ consists of the following transitions:

- $\delta_1 = (q_0, true, \emptyset, q_1)$, $\lambda(\delta_1) = \neg\tau$.
- $\delta_2 = (q_0, true, \{x_l^\tau := 0\}, q_1)$, $\lambda(\delta_2) = \tau$.
- $\delta_3 = (q_1, true, \{x_1^\tau := x_2^\tau, \ldots, x_{l-1}^\tau := x_l^\tau, x_l^\tau := 0\}, q_1)$, $\lambda(\delta_3) = \tau$.

Figure 4.4 shows $\mathcal{A}_\tau$. A run of $\mathcal{A}_\tau$ begins at time $t = 0$ in the initial state q_0. At the beginning all clock variables are inactive ($x_i^\tau = \perp$ for every $i \in \{1, \ldots, l\}$). If $\tau \in w^s[0]$ the run transitions to q_1 while making a clock assignment $x_l^\tau := 0$. In the other case, namely $\tau \notin w^s[0]$, the run transitions to q_1 without any clock transformations. The run stays in q_1 for every $t > 0$ such that $\tau \notin w^s[t]$. When $\tau \in w^s[t]$ the run transitions from q_1 to q_1 while shifting all clock variables one step

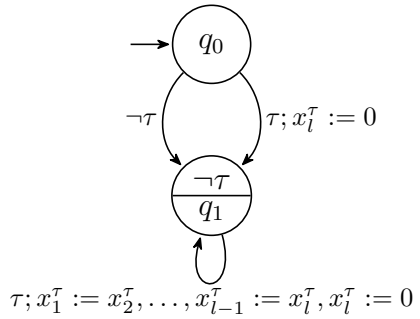

Figure 4.4: TA for punctual events $\mathcal{A}_\tau$

to the left ($x_1^\tau := x_2^\tau, \ldots, x_{l-1}^\tau := x_l^\tau$) and assigning $x_l^\tau := 0$. Thus at time t the clock x_l^τ contains the time elapsed since the latest event, x_{l-1}^τ contains the time elapsed since the second latest event, and so on, and by bounded event rate the time elapsed since each of the events that occurred in the time interval $[h-t, t]$ is held by a clock variable. Furthermore, if $\tau_{l'}^s \leq t < \tau_{l'+1}^s$, then for every $i \in \{1, \ldots, l\}$ such that $x_i^\tau \neq \perp$ it is true that $t - x_i^\tau = \tau_{l'-l+i}^s$

4.3.2 TA for atomic propositions

The TAs for atomic propositions are defined in a way similar to the TA for punctual events. It is assumed there is a known *bounded variability* k_{var} obeyed by every atomic proposition, meaning that for each atomic proposition there are at most k_{var} distinct intervals in each unit of time during which the proposition is true. Note that this follows from assuming a bound on event rate. The TA for an atomic proposition p needs to memorize all the subintervals of $[t-h, h]$ in which p holds continuously. Because of bounded variability there are at most $r = \left\lceil \frac{k_{var} \cdot h}{2} \right\rceil$ such subintervals. Formally the TA for a proposition p is $\mathcal{A}_p = \left(2^{\{p\}}, Q, \mathcal{C}, \lambda, I, \Delta, q_0, \{Q\}\right)$ where $Q = \{q_0, q_1, q_2\}$, $\mathcal{C} = \{x_1^p, y_1^p, \ldots, x_r^p, y_r^p\}$, $\lambda(q_1) = \neg p$, $\lambda(q_2) = p$, $I(q_1) = I(q_2) = true$ and transitions:

- $\delta_1 = (q_0, true, \emptyset, q_1)$, $\lambda(\delta_1) = \neg p$.
- $\delta_2 = (q_0, true, \{x_r^p := 0\}, q_2)$, $\lambda(\delta_2) = p$.
- $\delta_3 = (q_1, true, \{x_1^p := x_2^p, y_1^p := y_2^p, \ldots, x_{r-1}^p := x_r^p, y_{r-1}^p := y_r^p, x_r^p := 0, y_r^p := \perp\}, q_2)$, $\lambda(\delta_3) = p$.
- $\delta_4 = (q_2, true, \{y_r := 0\}, q_1)$, $\lambda(\delta_4) = \neg p$.

Figure 4.5 shows $\mathcal{A}_p$. A run of $\mathcal{A}_p$ begins in the initial state q_0. At first all clock variables are inactive ($x_i^p = \perp$ and $y_i^p = \perp$ for every $i \in \{1, \ldots, r\}$). If $p \in w^s[0]$ the

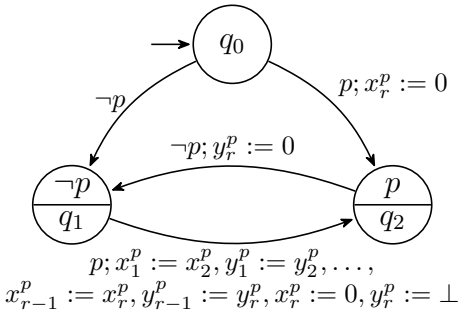

Figure 4.5: TA for atomic propositions $\mathcal{A}_p$.

run transitions to q_2 while making a clock assignment $x_l^p := 0$. In the other case, namely $p \notin w^s[0]$, the run transitions to q_1 without any clock transformations. Having entered q_1, the run remains in q_1 for every $t > 0$ such that $p \notin w^s[t]$. When $p \in w^s[t]$ the run transitions from q_1 to q_2 while shifting all clock variables one step to the left ($x_1^p := x_2^p, y_1^p := y_2^p, \ldots, x_{r-1}^p := x_r^p, y_{r-1}^p := y_r^p$) and assigning $x_r^p := 0$ and $y_r^p := \perp$. Having entered q_2, the run remains in q_2 for every t such that $p \in w^s[t]$. When $p \notin w^s[t]$ a transition is taken from q_2 to q_1 while assigning $y_r^p := 0$. Thus at time t clocks variables x_i^p and y_i^p ($i \in \{1, \ldots, r\}$) respectively hold the time elapsed since the beginning and end of each of the last r intervals in which p held, and by bounded variability of p every interval overlapping $[t-h, t]$ in which p held is memorized by such a clock pair.

Notice that this improves the construction in [25] by reducing the number of states from $2r$ to 2.

4.3.3 Combined TA for events and propositions

Let $\mathcal{A}_{MEM} = \mathcal{A}_\tau \| \mathcal{A}_{p_1} \| \ldots \| \mathcal{A}_{p_m}$ where $\{p_1, \ldots, p_m\} = AP$.

4.4 Predicates

We define predicates $C_\psi^\alpha(i)$ ($\alpha \in \{0, 1\}$) that are valid clock constraints with each subformula ψ inductively on the structure of φ. If ψ is an unbounded temporal operator we associate a DTA D_ψ^α with $C_\psi^\alpha(i)$. Let $l'(t)$ be the index of s at time t satisfying $\tau^s_{l'(t)} \leq t < \tau^s_{l'(t)+1}$ and let $i'(t,i) = l'(t) - l + i$. Given that $-x_i^\tau \in [-h + pas(\psi), -fut(\psi)]$ at time t, then:

- the predicate $C_\psi^0(i)$ is satisfied at time t iff $s, i'(t,i) \models \psi$ and
- the predicate $C_\psi^1(i)$ is satisfied at time t iff $\overline{s}, i'(t,i) \models \psi$.

The predicates are defined inductively for every subformula ψ of φ by:

- $\psi = p$. $C_p^\alpha(i) = \left(\bigvee_j -x_j^p \leq -x_i^\tau < -y_j^p\right) \vee C_\top^\alpha(i)$ where $C_\top^\alpha(i)$ is defined at the end of this subsection.
- $\psi = \neg\psi_1$. $C_{\neg\psi}^0(i) = \neg C_{\psi_1}^1(i)$. $C_{\neg\psi_1}^1(i) = \neg C_{\psi_1}^0(i)$.
- $\psi = \psi_1 \vee \psi_2$. $C_{\psi_1\vee\psi_2}^\alpha(i) = C_{\psi_1}^\alpha(i) \vee C_{\psi_2}^\alpha(i)$.
- $\psi = \psi_1 \mathcal{U}_I \psi_2$, $I = (a, b)$ or $I = [b, b]$.

$$C_{\psi_1\mathcal{U}_I\psi_2}^\alpha(i) = \bigvee_{j=i+1}^{l} \left[-x_j^\tau \in -x_i^\tau + I \wedge C_{\psi_2}^\alpha(j) \wedge \bigwedge_{k=i+1}^{j-1} C_{\psi_1}^\alpha(k) \right]$$

- $\psi = \psi_1 \mathcal{S}_I \psi_2$, $I = (a, b)$ or $I = [b, b]$.

$$C_{\psi_1\mathcal{S}_I\psi_2}^\alpha(i) = C_\top^\alpha(i) \vee \bigvee_{j=1}^{i-1} \left[-x_j^\tau \in -x_i^\tau - I \wedge C_{\psi_2}^\alpha(j) \wedge \bigwedge_{k=j+1}^{i-1} C_{\psi_1}^\alpha(k) \right]$$

  where $C_\top^\alpha(i)$ is defined at the end of this subsection.
- $\psi = \psi_1 \mathcal{U}_{(a,\infty)} \psi_2$ is defined in Section 4.5.2.
- $\psi = \psi_1 \mathcal{S}_{(a,\infty)} \psi_2$ is defined in Section 4.5.1.

The two "helper" predicates $C_\top^\alpha(i)$ have the property that if $-x_i^\tau \in [-h, 0]$ at time t, then:

- the predicate $C_\top^0(i)$ is satisfied at time t iff $\rho_{i'(t,i)}^s = \top$ and is defined by $C_\top^0(i) = \bigvee_j -x_j^\top \leq -x_i^\tau < -y_j^\top$, and
- the predicate $C_\top^1(i)$ is satisfied at time t iff $\rho_{i'(t,i)}^s = \bot$ and is defined by $C_\top^1(i) = \bigvee_j -x_j^\bot \leq -x_i^\tau - y_j^\bot$.

4.5 Unbounded formulas

In general the truth of an unbounded temporal operator $\psi = \psi_1 \mathcal{U}_{(a,\infty)} \psi_2$ or $\psi = \psi_1 \mathcal{S}_{(a,\infty)} \psi_2$ cannot be decided by only looking within a bounded sliding time window. It is however possible to detect (or guess in the case of future temporal operators) the beginnings and ends of the sequences of consecutive events for which ψ holds. We use a specially constructed DTA D_ψ^α ($\alpha \in \{0, 1\}$) with output p_ψ^α which keeps track of whether a specific time offset relative to the sliding window lies inside such an interval and outputs p_ψ^α if it does and $\neg p_\psi^\alpha$ otherwise.

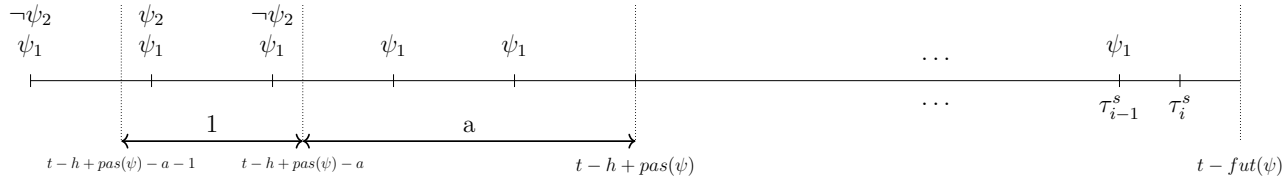

Figure 4.6: The beginning and end of every interval where $\psi = \psi_1 \mathcal{S}_{(a,\infty)} \psi_2$ holds can be observed by remembering when $\psi_1 \mathcal{S}_{(a,a-1)} \psi_2$ held in the past.

4.5.1 Unbounded since

In this section we construct a DTA D^α_ψ for subformulas $\psi = \psi_1 \mathcal{S}_{(a,\infty)} \psi_2$ which outputs p^α_ψ at time t_1 iff

$$\begin{aligned} \exists j : \tau^s_j \in (-\infty, t_1 - h + pas(\psi) - a - 2].s^\alpha, j \models \psi_2 \wedge \\ \forall k : \tau^s_k \in (\tau^s_j, t_1 - h + pas(\psi) - 1].s^\alpha, k \models \psi_1 \end{aligned} \tag{2}$$

where $s^0 = s$ and $s^1 = \overline{s}$.

The invariant (2) allows us to "stitch" together the truth of $s^\alpha, i \models \psi$ in the following way: For every event i with $\tau^s_i \in [t_1 - h + pas(\psi), t_1 - fut(\psi)]$ it is true that $s^\alpha, i \models \psi$ iff either:

- (2) holds at t_1 and every for event j with $\tau^s_j \in (t_1 - h + pas(\psi) - 1, \tau^s_i)$ it holds that $s^\alpha, j \models \psi_1$, or
- there is some $j < i$ with $\tau^s_j \in (t_1 - h + pas(\psi) - a - 2, \tau^s_i - a)$ such that $s^\alpha, j \models \psi_2$ and every k with $i < k < j$ satisfies $s^\alpha, k \models \psi_1$.

If it is known whether (2) holds or not at t_1 it is sufficient to only consider a bounded time window to tell whether ψ is satisfied.

The precise form of the intervals in (2) with the additional "look-backs" -1 and -2 are picked to ensure that (2) holds in left-open right-closed time intervals (and the same is true for its negation), which simplifies the proof of correctness by ensuring that every interval has a first time point.

The construction of D^α_ψ relies on the following three insights about the intervals when (2) holds:

- The invariant (2) does not hold at the first event, because there are no events preceding it where ψ_2 could be true.
- Every interval where (2) holds begins with a time t_1 where there is an event j such that $\tau^s_j \in (t_1 - h + pas(\psi) - a - 3, 1_0 - h + pas(\psi) - a - 2]$ for which it holds that $s^\alpha, j \models \psi_2$, and for every k such that $\tau^s_k \in (\tau^s_k, t_1 - h + pas(\psi) - 1]$ it holds that $s^\alpha, k \models \psi_1$.

- Every interval where (2) holds ends with a time t_1 where there is some k such that $\tau_k^s \in (t_1 - h + pas(\psi) - a - 2, t_1 - h + pas(\psi) - 1]$ for which it holds that $s^\alpha, k \not\models \psi_1$.

These requirements are readily captured by a DTA with two states (Figure 4.7), the first one, q_1, representing (2) holding at t_1 and the second one, q_2 representing (2) not holding at t_1. Because (2) cannot hold at time $t = 0$ it follows that q_1 is the initial state. The beginnings and ends of intervals in which (2) holds are then readily expressed by staying conditions and transition guards. More concretely, D_ψ^α has two states:

- Initial state q_1 with output $\gamma(q_1) = \neg p_\psi^\alpha$ and staying condition

$$I_{q_1}^\alpha = \bigwedge_{j=1}^{l} \Big\{ -x_j^\tau \in (-h + pas(\psi) - a - 3, -h + pas(\psi) - a - 2] \implies \Big[\neg C_{\psi_2}^\alpha(j) \vee \bigvee_{k=j+1}^{l} \Big(-x_k^\tau \in (-x_j^\tau, -h + pas(\psi) - 1] \wedge \neg C_{\psi_1}^\alpha(k) \Big) \Big] \Big\}$$

- and state q_2 with output $\gamma(q_2) = p_\psi^\alpha$ and staying condition

$$I_{q_2}^\alpha = \bigwedge_{k=1}^{l} \Big(-x_k^\tau \in (-h + pas(\psi) - a - 2, -h + pas(\psi) - 1] \implies C_{\psi_1}^\alpha(k) \Big)$$

There are transitions connecting q_1 with q_2 and q_2 back with q_1:

- $\delta_2 = (q_2, g_1^\alpha, \neg p_\psi^\alpha, q_1)$ where the transition guard is

$$g_1^\alpha = \bigvee_{k=1}^{l} \Big(-x_k^\tau \in (-h + pas(\psi) - a - 2, -h + pas(\psi) - 1] \wedge \neg C_{\psi_1}^\alpha(k) \Big)$$

- and $\delta_1 = (q_1, g_2^\alpha, p_\psi^\alpha, q_2)$ where the transition guard is

$$g_2^\alpha = \bigvee_{j=1}^{l} \Big[-x_j^\tau \in (-h + pas(\psi) - a - 3, -h + pas(\psi) - a - 2] \wedge C_{\psi_2}^\alpha(j) \wedge \bigwedge_{k=j+1}^{l} \Big(-x_k^\tau \in (-x_j^\tau, -h + pas(\psi) - 1] \implies C_{\psi_1}^\alpha(k) \Big) \Big]$$

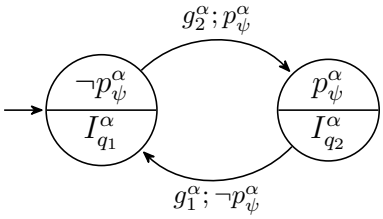

Figure 4.7: DTA for unbounded since D_ψ^α

It should be clear from the above definitions that $g_1^\alpha = \neg I_{q_2}^\alpha$ and $g_2^\alpha = \neg I_{q_1}^\alpha$.

Finally we define the predicate for ψ by:

$$\begin{aligned} C_\psi^\alpha(i) =& C_\top^\alpha(i) \vee \\ & \left[p_\psi^\alpha \wedge \bigwedge_{k=1}^{i-1} \Big(-x_k^\tau \in (-h + pas(\psi) - 1, -x_i^\tau) \implies C_{\psi_1}^\alpha(k) \Big) \right] \vee \\ & \left[\bigvee_{j=1}^{i-1} \left(-x_j^\tau \in (-h + pas(\psi) - a - 2, -x_i^\tau - a) \wedge C_{\psi_2}^\alpha(j) \wedge \bigwedge_{k=j+1}^{i-1} C_{\psi_1}^\alpha(k) \right) \right] \end{aligned}$$

4.5.2 Unbounded until

This part of the construction is adapted from [25]. In this section we construct a DTA D_ψ^α for subformulas $\psi = \psi_1 \mathcal{U}_{(a,\infty)} \psi_2$ which outputs p_ψ^α at time t_1 iff

$$\begin{aligned} \exists j : \tau_j^s \in (t_1 - fut(\psi) + a, \infty).s^\alpha, j \models \psi_2 \\ \wedge \forall k : \tau_k^s \in (t_1 - fut(\psi), \tau_j^s).s^\alpha, k \models \psi_1 \end{aligned} \tag{3}$$

where $s^0 = s$ and $s^1 = \overline{s}$. Just like in Section 4.5.1, we choose the invariant (3) in a way that allows us to "stitch" together the truth of $s^\alpha, i \models \psi$ while memorizing only propositions in only a bounded time window.

The construction of D_ψ^α for unbounded until is necessarily more complicated than the one for unbounded since. This is because in the case of unbounded since the beginning of an interval where (2) holds can be directly observed by memorizing events and propositions in a bounded time window. Whether the invariant (3) holds at time t depends on the occurrence of an event j with $s^\alpha, j \models \psi_2$ where τ^j may be arbitrarily bigger than t. It is thus necessary to guess whether (3) holds or not at time t.

The key to the construction of D_ψ^α is the realization that there are three distinct classes of knowledge if observing a bounded time window at time t:

1. It can be directly observed that (3) holds at time t because there is an event j such that $\tau_j^s \in (t - fut(\psi) + a, t - fut(\psi) + a + 1)$ for which it holds that $s^\alpha, j \models \psi_2$ and for every k such that $\tau_k^s \in (t - fut(\psi), \tau_j^s)$ it holds that $s^\alpha, k \models \psi_1$.

2. It can be directly observed that (3) does not hold at time t because every event j such that $\tau_j^s \in (t - fut(\psi) + a, t - fut(\psi) + a + 1)$ for which it holds that $s^\alpha, j \models \psi_2$ there is an event k such that $\tau_k^s \in (t - fut(\psi) + a, \tau_j^s)$ for which it holds that $s^\alpha, k \not\models \psi$.

3. It cannot be directly observed whether (3) does hold or does not hold at time t because every event j with $\tau_j^s \in (t - fut(\psi) + a, t - fut(\psi) + a + 1)$ it holds that $s^\alpha, j \not\models \psi_2$ and for every k such that $\tau_k^s \in (t - fut(\psi), t - fut(\psi) + a + 1)$ it holds that $s^\alpha, k \models \psi_1$.

Thus (3) holding at time t can be directly observed while memorizing only a bounded time window in two out of the three classes. The third class is the source of complication. Note however that if time t occurs in an interval of the third class and that interval is followed by a time interval of the first class, then (3) held at time t, and if time t occurred in an interval of the third class and is followed by a time interval of the second class, then (3) did not hold at time t, and if time t occurred in a time interval of the third which also happened to be the last interval, then (3) did not hold at time t.

These ideas are captured by a DTA with five states (Figure 4.8), the first one, q_0, being an initial state waiting for some event to enter the time window where the predicates for ψ_1 and ψ_2 to are valid. There are outgoing transitions from q_0 to all other states. States q_2 and q_4 respectively represent the first and second class of observations and have outgoing transitions to all other states other than q_0. The state q_1 represents the third class of observations with a commitment to a guess that the next class is the first class, and thus that (3) holds, and this is represented by q_1 only having a single outgoing transition to q_2. Thus a run that entered q_1 at time t followed by an observation of the third class at time t' is terminated. In a similar way, state q_3 represents the third class of observations with a commitment to a guess that the next class is the second class, and thus that (3) does not hold, and this is represented by q_3 having a single outgoing transition to q_4. Both guesses are made non-deterministically, but since we only consider infinite runs we can assume the right guess was made every time.

More concretely D_ψ^α has the following states:

- Initial state q_0 with output $\gamma(q_0) = o$ and staying condition $I_{q_0}^\alpha = \bigwedge_{i=1}^{l} x_i^\tau < fut(\psi)$.

- State q_1 is the only *unfair state* with output $\gamma(q_1) = p^\alpha_\psi$ and staying condition

$$I^\alpha_{q_1} = \bigwedge_{k=1}^{l} \Big(-x^\tau_k \in (-fut(\psi), -fut(\psi)+a+1) \implies C^\alpha_{\psi_1}(k) \Big) \wedge \bigwedge_{j=1}^{l} \Big(-x^\tau_j \in (-fut(\psi)+a, -fut(\psi)+a+1) \implies \neg C^\alpha_{\psi_2}(j) \Big)$$

  By unfair we mean that a run that stays forever in q_1 is not accepting.

- State q_2 with output $\gamma(q_2) = p^\alpha_\psi$ and staying condition

$$I^\alpha_{q_2} = \bigvee_{j=1}^{l} \Big[-x^\tau_j \in (-fut(\psi)+a, -fut(\psi)+a+1) \wedge C^\alpha_{\psi_2}(j) \wedge \bigwedge_{k=1}^{j-1} \Big(-x^\tau_k \in (-fut(\psi), -x^\tau_j) \implies C^\alpha_{\psi_1}(k) \Big) \Big]$$

- State q_3 with output $\gamma(q_3) = \neg p^\alpha_\psi$ and staying condition $I^\alpha_{q_3} = I^\alpha_{q_1}$.

- State q_4 with output $\gamma(q_4) = \neg p^\alpha_\psi$ and staying condition

$$I^\alpha_{q_4} = \bigvee_{k=1}^{l} \Big[-x^\tau_k \in (-fut(\psi), -fut(\psi)+a+1) \wedge \neg C^\alpha_{\psi_1}(k) \wedge \bigwedge_{j=1}^{k-1} \Big(-x^\tau_j \in (-fut(\psi)+a, -x^\tau_k) \implies \neg C^\alpha_{\psi_2}(k) \Big) \Big]$$

The transition guards are directly associated with the destination state of the transition so that g_1 is associated with q_1, g_2 with q_2 and so on:

- $g^\alpha_1 = g^\alpha_3 = I^\alpha_{q_1} \wedge \bigwedge_{j=1}^{l} \big[-x^\tau_j = -fut(\psi)+a+1 \implies \big(C^\alpha_{\psi_1}(j) \wedge \neg C^\alpha_{\psi_2}(j) \big) \big]$.

- $g^\alpha_2 = I^\alpha_{q_2} \vee \Big[I^\alpha_{q_1} \wedge \bigvee_{j=1}^{l} \big(-x^\tau_j = -fut(\psi)+a+1 \wedge C^\alpha_{\psi_2}(j) \big) \Big]$.

- $g^\alpha_4 = I^\alpha_{q_4} \vee \Big[I^\alpha_{q_3} \wedge \bigvee_{j=1}^{l} \big(-x^\tau_j = -fut(\psi+a+1 \wedge \neg C^\alpha_{\psi_1}(j) \wedge \neg C^\alpha_{\psi_2}(j) \big) \Big]$.

Note that $g_1 = g_3$. The transition guards correspond to the staying condition holding in an open interval following the current time t. This simplifies the proof of correctness by ensuring that the run stays in each state for a non-singular amount of time. Although we do not use it here, it also means D^α_ψ is determinizable using the scheme given in [25].

The transitions of D^α_ψ are:

- $\left(q_0, \bigvee_{i=1}^{l} (x^\tau_i = fut(\psi)) \wedge g^\alpha_1, p^\alpha_\psi, q_1\right)$
- $\left(q_0, \bigvee_{i=1}^{l} (x^\tau_i = fut(\psi)) \wedge g^\alpha_2, p^\alpha_\psi, q_2\right)$
- $\left(q_0, \bigvee_{i=1}^{l} (x^\tau_i = fut(\psi)) \wedge g^\alpha_3, \neg p^\alpha_\psi, q_3\right)$
- $\left(q_0, \bigvee_{i=1}^{l} (x^\tau_i = fut(\psi)) \wedge g^\alpha_4, \neg p^\alpha_\psi, q_4\right)$
- $\left(q_1, g^\alpha_2, p^\alpha_\psi, q_2\right)$
- $\left(q_2, g^\alpha_1, p^\alpha_\psi, q_1\right)$
- $\left(q_2, g^\alpha_3, \neg p^\alpha_\psi, q_3\right)$
- $\left(q_2, g^\alpha_4, \neg p^\alpha_\psi, q_4\right)$
- $\left(q_3, g^\alpha_4, \neg p^\alpha_\psi, q_4\right)$
- $\left(q_4, g^\alpha_1, p^\alpha_\psi, q_1\right)$
- $\left(q_4, g^\alpha_2, p^\alpha_\psi, q_2\right)$
- $\left(q_4, g^\alpha_3, \neg p^\alpha_\psi, q_3\right)$

Finally we define the predicate for ψ by:

$$C^\alpha_\psi(i) = \left[-x^\tau_i \leq -fut(\psi) \wedge p^\alpha_\psi \wedge \bigwedge_{k=i+1}^{l} \Big(-x^\tau_k \in (-x^\tau_i, -fut(\psi)] \implies C^\alpha_{\psi_1}(k)\Big)\right] \vee$$
$$\Big[-x^\tau_i < -fut(\psi) \wedge \bigvee_{j=i+1}^{l} \Big(-x^\tau_j \in (-x^\tau_i + a, -fut(\psi) + a] \wedge C^\alpha_{\psi_2}(j)$$
$$\wedge \bigwedge_{k=i+1}^{j-1} C^\alpha_{\psi_1}(k) \Big) \Big]$$

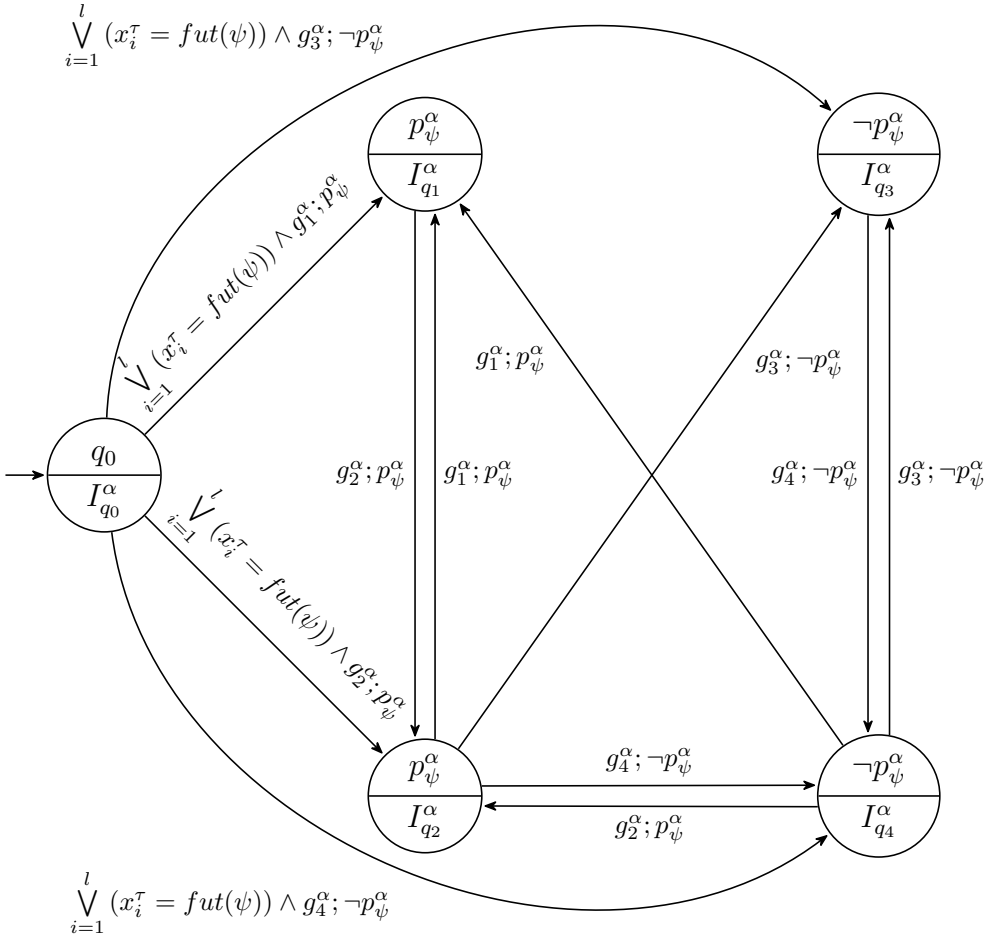

Figure 4.8: DTA for unbounded until D_ψ^α

4.6 Construction of $\mathcal{A}_\varphi$

Let $\psi_1, \ldots, \psi_n$ be all the unbounded temporal operator subformulas of φ such that ψ_i and ψ_j have $i < j$ if ψ_i is a sub formula of ψ_j, and let $\alpha_1, \ldots, \alpha_n$ be the $\alpha_i \in \{0, 1\}$ which appears as the predicate for $C^{\alpha_i}_{\psi_1}(i)$ subformula ψ_i in $C^0_\varphi(i)$ as constructed inductively in Section 4.4. Form a DTA $D = D^{\alpha_1}_{\psi_1} \otimes \cdots \otimes D^{\alpha_n}_{\psi_n}$ of all unbounded temporal operator subformulas. D has output alphabet $\Gamma = 2^{\{p^{\alpha_1}_{\psi_1}, \ldots, p^{\alpha_n}_{\psi_1}\}}$.

We now construct a DTA B for the formula φ (Figure 4.9) with the following states and transitions:

- Initial state q_0 with staying condition $I_{q_0} = \bigwedge_{i=1}^{l} x_i^\tau < fut(\varphi)$ and outgoing transitions $\delta_1 = (q_0, g_{accept}, o, q_{accept})$ where

$$g_{accept} = \bigvee_{i=1}^{l} \Big[(x_i^\tau = fut(\varphi)) \wedge C^0_\varphi(i) \Big]$$

  and $\delta_2 = (q_0, g_{reject}, o, q_{reject})$ where

$$g_{reject} = \bigvee_{i=1}^{l} \Big[(x_i^\tau = fut(\varphi)) \wedge \neg C^0_\varphi(i) \Big]$$

  and o is some arbitrary output symbol.

- Accepting sink state q_{accept} with staying condition $I_{q_{accept}} = true$.
- Rejecting sink state q_{reject} with staying condition $I_{q_{reject}} = true$.

A run of B begins in the initial state q_0 and remains there as long as every clock variable satisfies $x_i^\tau < fut(\varphi)$ ($i \in \{1, \ldots, l\}$). Note that $C^0_\varphi(i)$ is defined for every i such that $x_i^\tau \in [h - pas(\varphi), fut(\varphi)] = [fut(\varphi), fut(\varphi)] = \{fut(\varphi)\}$ (the singleton set containing $fut(\varphi)$) and the first clock variable to reach $x_i^\tau = fut(\varphi)$ must correspond to the time elapsed since the first event. Thus the run enters q_{accept} iff $s, 1 \models \varphi$ and enters q_{reject} iff $s, 1 \not\models \varphi$.

The TA for φ is $\mathcal{A}_\varphi = \mathcal{A}_{MEM} \otimes D_\varphi$ where $D_\varphi = D \otimes B$.

Example

We now return to our example formula (1) from Section 4.2. For ease of reading we repeat (1) again here:

$$\begin{aligned} \varphi &= \Box_{(0,\infty)} \big\{ \big(p \, \mathcal{S}_{(1,10)} q \big) \implies \big(r \, \mathcal{U}_{(5,10)} r \vee s \big) \big\} \\ &= \neg \big\{ true \, \mathcal{U}_{(0,\infty)} \neg \big(\big[p \, \mathcal{S}_{(1,10)} q \big] \vee \neg \big[r \, \mathcal{U}_{(5,10)} s \big] \big) \big\} \end{aligned}$$

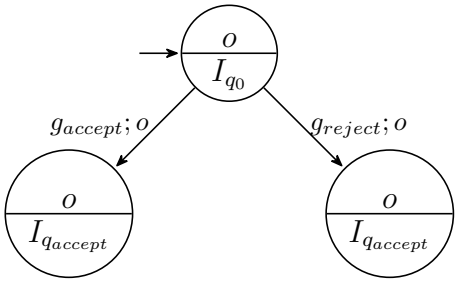

Figure 4.9: DTA for φ

This formula has $pas(\varphi) = 10$ and $fut(\varphi) = 12$, and thus $h = 22$. In the following we will let $\psi = true\,\mathcal{U}_{(0,\infty)}\psi_2$ where $\psi_2 = \neg\left(\left[p\,\mathcal{S}_{(1,10)}q\right] \vee \neg\left[r\,\mathcal{U}_{(5,10)}s\right]\right)$ so that (1) becomes $\neg\psi$.

We now demonstrate how the construction translates φ into a TA $\mathcal{A}_\varphi$ in hope providing a concrete example will ease understanding of the construction. For this formula we have $AP = \{p, q, r, s\}$, each with TA $\mathcal{A}_p$, $\mathcal{A}_q$, $\mathcal{A}_r$, $\mathcal{A}_s$ and $\mathcal{A}_{MEM} = \mathcal{A}_\tau \| \mathcal{A}_p \| \mathcal{A}_q \| \mathcal{A}_r \| \mathcal{A}_s$ as defined in Section 4.3.

The unbounded until subformula ψ has associated DTA D^1_ψ and proposition p^1 as defined in Section 4.5.2. When the unbounded until is of the form $\text{true}\,\mathcal{U}_{(a,\infty)}\psi_2$ state q_4 becomes unreachable as depicted in Figure 4.10.

Inserting the predicates from Section 4.4 and Section 4.5.2 we get:

- Predicate $C^0_{\neg\psi}(i) = \neg C^1_\psi(i)$.
- Predicate

$$C^1_\psi(i) =$$
$$\left[-x^\tau_i \leq -12 \wedge p^1_\psi \wedge \bigwedge_{k=i+1}^{l} \Big(-x^\tau_k \in (-x^\tau_i, -12] \implies \text{true} \Big)\right] \vee$$
$$\Big[-x^\tau_i < -12 \wedge \bigvee_{j=i+1}^{l} \Big(-x^\tau_j \in (-x^\tau_i, -12] \wedge C^1_{\psi_2}(j)$$
$$\wedge \bigwedge_{k=i+1}^{j-1} \text{true} \Big) \Big] =$$
$$\left[-x^\tau_i \leq -12 \wedge p^1_\psi\right] \vee \left[-x^\tau_i < -12 \wedge \bigvee_{j=i+1}^{l} \Big(-x^\tau_j \in (-x^\tau_i, -12] \wedge C^1_{\psi_2}(j) \right]$$

- Predicate $C^1_{\psi_2}(i) = \neg C^0_{\neg\left[p\,\mathcal{S}_{(a,b)}q\right] \vee r\,\mathcal{U}_{(c,d)}s}(i)$.

- Predicate $C^0_{\neg\left[p\,\mathcal{S}_{(a,b)}q\right]\vee r\,\mathcal{U}_{(c,d)}s}(i) = C^0_{\neg\left[p\,\mathcal{S}_{(a,b)}q\right]}(i) \vee C^0_{r\,\mathcal{U}_{(c,d)}s}(i)$.

- Predicate $C^0_{\neg\left[p\,\mathcal{S}_{(a,b)}q\right]}(i) = \neg C^1_{p\,\mathcal{S}_{(a,b)}q}(i)$.

- Predicate

$$C^1_{p\,\mathcal{S}_{(1,10)}q}(i) = C^1_{\top}(i) \vee \bigvee_{j=1}^{i-1} \left[-x^\tau_j \in -x^\tau_i - (5,10) \wedge C^1_q(j) \wedge \bigwedge_{k=j+1}^{i-1} C^1_p(k) \right]$$

- Predicate

$$C^0_{r\,\mathcal{U}_{(1,10)}s}(i) = \bigvee_{j=i+1}^{l} \left[-x^\tau_j \in -x^\tau_i + (5,10) \wedge C^0_q(j) \wedge \bigwedge_{k=i+1}^{j-1} C^0_p(k) \right]$$

Note that true makes the predicate for C^1_ψ contract.

Finally $\mathcal{A}_{\neg\psi}$ is constructed following the procedure described in Section 4.6 with $\mathcal{A}_{\neg\psi} = \mathcal{A}_{MEM} \otimes D^{\neg\psi}$ where $D^{\neg\psi} = D^1_\psi \otimes B$ and the transitions to the accepting and rejecting sink states of B have guards:

- $g_{accept} = \bigvee_{i=1}^{l} \left[(x^\tau_i = 12) \wedge C^0_{\neg\psi}(i)\right]$

- $g_{reject} = \bigvee_{i=1}^{l} \left[(x^\tau_i = 12) \wedge \neg C^0_{\neg\psi}(i)\right]$

Clearly g_{accept} and g_{reject} are mutually exclusive and a run of B will transition to one of the sink state there is some i $(1 \leq 1 \leq l)$ such that $x^\tau_i = 12$. Since $x^\tau_i = 12$ the predicate $C^0_{\neg\psi} = \neg C^1_\psi(i)$ contracts to $\neg p^1_\psi$ meaning the choice of sink state is determined by the negation of the output of D^1_ψ at time $t = \tau^s_1 + 12$.

Let ξ^1_ψ be an accepting run of D^1_ψ and the state of ξ^1_ψ at time t:

- If the output of ξ^1_ψ at time t is $\neg p^1_\psi$ the sink state is q_{reject} (thus the run of $\mathcal{A}_{\neg\psi}$ is rejecting) and ξ^1_ψ is in q_3 or is taking a transition to q_3 and subsequently remains in q_3 state forever.

- Conversely if the output of ξ^1_ψ at time t is p^1_ψ the sink state is q_{accept} (thus the run of $\mathcal{A}_\psi$ is accepting) and ξ^1_ψ is in or is taking a transition to one of q_1 or q_2, but because q_1 is the only unfair state D^1_ψ and because ξ^1_ψ is accepting the run must eventually enter q_2.

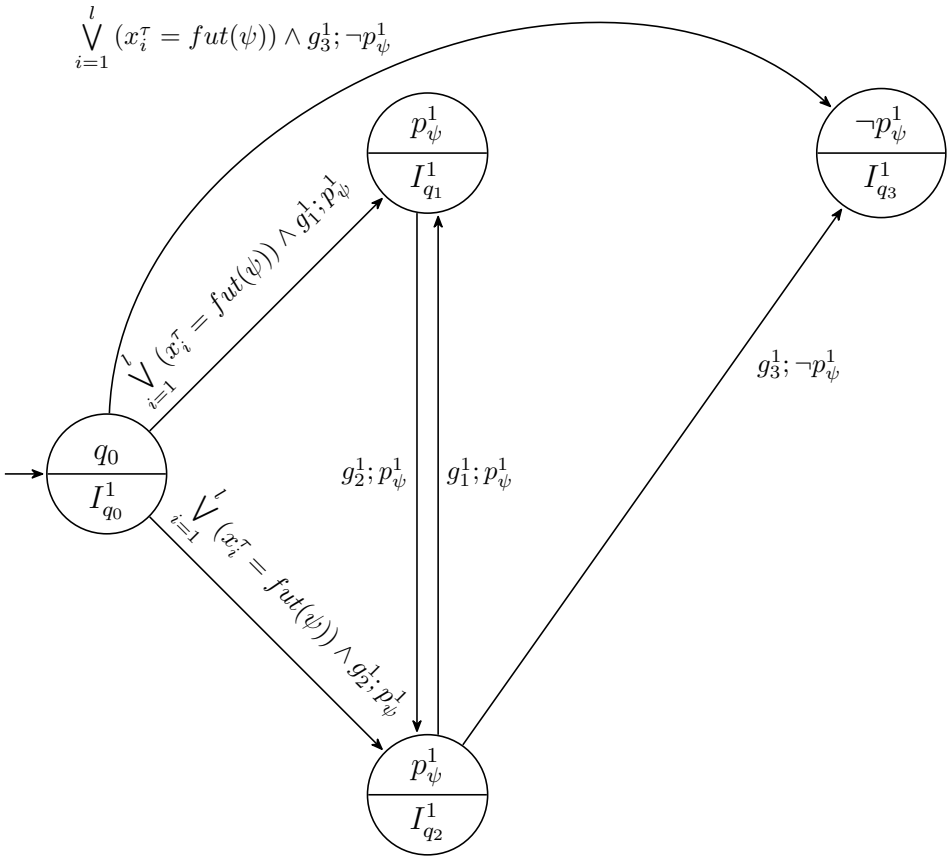

Figure 4.10: DTA for $\psi = \text{true}\mathcal{U}_{(a,\infty)}\psi_2$, D_ψ^1

4.7 Correctness and complexity analysis

It remains to show that there is a correspondence between signals accepted by the construction and words accepted by the target semantics, or more formally that $w^s \in \mathcal{A}_\varphi$ iff $s, 1 \models \varphi$. Lemma 5 and Lemma 6 establish the right and left implications of the equivalence, but with somewhat stronger statements. From these ?THM? **??** and Corollary 1, which are the main results of the section, follow directly.

Lemma 5. *Assume there is an run ξ of $\mathcal{A}_\varphi$ on w^s which is accepting. Then given that this run is in some state at t_1, every $i \in \{1, \ldots, l\}$ such that $i'(t_1, i) \leq l'(t_1)$ and $x_i^\tau \neq \bot$ satisfies $\tau^s_{i'(t_1,i)} = t_1 - x_i^\tau$, and for every subformula ψ of φ if $-x_i^\tau \in [-h+pas(\psi), -fut(\psi)]$, the predicate $C^0_\psi(i)$ is satisfied iff $s, i'(t_1, i) \models \psi$ and $C^1_\psi(i)$ is satisfied iff $\overline{s}, i'(t_1, i) \models \psi$.*

Proof. The proof proceeds by induction on the formula φ. Let $h = pas(\varphi)+fut(\varphi)$ and let D^α_ψ be the DTA for the subformula ψ of one of the forms $\psi = \psi_1 \mathcal{U}_{(a,\infty)}$ or $\psi = \psi_1 \mathcal{S}_{(a,\infty)} \psi_2$.

- $\tau^s_{i'(t_1,i)} = t_1 - x_i^\tau$. It can be seen from definition that the TA for punctual events takes a discrete step at time t iff $\tau \in w^s[t]$, and by Definition 5 $\tau \in w^s[t]$ iff $t = \tau^s_j$ for some $j > 0$. During such a discrete step of the TA, all clock indices are shifted left by one position, so that $x_j^\tau = x_{j+1}^{'\tau}$ is the value of clock j directly following the discrete step iff $x_{j+1}^{'\tau}$ was the value of clock $i+1$ directly before the step ($j \in \{1, \ldots, l-1\}$). At time t_1, we have by definition $\tau^s_{l'(t_1)} \leq t_1 < \tau^s_{l'(t_1)+1}$, and the value of clock x^τ_{l-j} ($j = 0, \ldots, l-1$) originates from a clock restarted at time $\tau^s_{l'(t_1)-j}$ and since incremented continuously. It follows that $x^\tau_{l-j} = t_1 - \tau^s_{l'(t_1)-j}$. Substituting $i = l - j$ then gives $t_1 - x_i^\tau = \tau^s_{l'(t_1)-l+i} = \tau^s_{i'(t_1,i)}$.

- $\top \in \rho$. If $C^0_\top(i)$ is satisfied at t_1, then there is a clock pair j satisfying $-x_j^\top \leq -x_i^\tau < -x_j^\top$. Then at $\tau^s_{i'(t_1,i)}$, $-x_j^\top \leq 0$ and $-y_j^\top > 0$, that is, the TA for $\top$ was in state q_2 observing $\top \in w^s[\tau^s_{i'(t_1,i)}]$, implying $\rho^s_{i'(t_1,i)} = \top$.

  Conversely, if $C^0_\top(i)$ is not satisfied at t_1, then for every clock pair j, either $-x_j^\top > -x_i^\tau$ or $-x_i^\tau \geq -y_j^\top$. Then at $\tau^s_{i'(t_1,i)}$, $-x_j^\top > 0$ or $-y^\top \leq 0$, that is the TA for $\top$ was in state q_1 observing $\top \notin w^s[\tau^s_{i'(t_1,t)}]$, implying $\rho^s_{i'(t_1,i)} \neq \top$.

  The proof for $C^1_\top(i)$ is similar.

- $\psi = p$. If $C^0_p(i)$ is satisfied at t_1, then either $C^0_\top(i)$ is satisfied, implying $\rho^s_{i'(t_1,i)} = \top$ and thus $s, i'(t_1, i) \models p$, or there is a clock pair j satisfying $-x_j^p \leq -x_i^\tau < -y_j^p$. Then at $\tau^s_{i'(t_1,t)}$, $-x_j^p \leq 0$ and $-y_j^p > 0$, that is, the TA

for p was in state q_2 observing $p \in w^s[\tau^s_{i'(t_1,i)}]$, implying $p \in \rho^s_{i'(t_1,i)}$ and thus $s, i'(t_1, i) \models p$.

Conversely, if $C^0_\top(i)$ is not satisfied t_1, then for every clock pair j, either $-x^p_j > -x^\tau_i$ or $-x^\tau_i \geq -y^p_j$. Then at $\tau^s_{i'(t_1,i)}$, $-x^p_j > 0$ or $-y^p_j \leq 0$, that is the TA for p was in state q_1 observing $p \notin w^s[\tau^s_{i'(t_1,t)}]$, implying $p \notin \rho^s_{i'(t_1,t)}$. Similarly $C^0_\top(i)$ was not satisfied at t_1, implying $\rho^s_{i'(t_1,t)} \neq \top$. Combining, we have $s, i'(t_1, i) \not\models p$.

The proof for $C^1_p(i)$ is similar.

- $\psi = \neg\psi_1$. If $C^0_{\neg\psi_1}(i)$ is satisfied at t_1, then $C^1_{\psi_1}(i)$ is not satisfied at t_1, and by induction $\overline{s}, i'(t_1, i) \not\models \psi_1$, implying $s, i'(t_1, i) \models \neg\psi_1$.

  Conversely, if $C^0_{\neg\psi_1}(i)$ is not satisfied at t_1, then $C^1_{\psi_1}(i)$ is satisfied at t_1, and by induction $\overline{s}, i'(t_1, i) \models \psi_1$, implying $s, i'(t_1, i) \not\models \neg\psi_1$.

  The proof for $C^1_{\neg\psi(i)}$.

- $\psi = \psi_1 \vee \psi_2$. If $C^0_{\psi_1\vee\psi_2}(i)$ is satisfied at t_1, then either $C^0_{\psi_1}(i)$ or $C^0_{\psi_2}(i)$ is satisfied at t_1, and by induction either $s, i'(t_1, i) \models \psi_1$ or $s, i'(t_1, i) \models \psi_2$, implying $s, i'(t_1, i) \models \psi_1 \vee \psi_2$.

  Conversely, if $C^0_{\psi_1\vee\psi_2}(i)$ is not satisfied at t_1, then neither of $C^0_{\psi_1}(i)$ and $C^0_{\psi_2}(i)$ are satisfied at t_1, and by induction $s, i'(t_1, i) \not\models \psi_1$ and $s, i'(t_1, t) \not\models \psi_2$, implying $s, i'(t_1, i) \not\models \psi_1 \vee \psi_2$.

  The proof for $C^1_{\psi_1\vee\psi_2}(i)$ is similar.

- $\psi = \psi_1 \mathcal{S}_I \psi_2$ where $I = (a, b)$ or $I = [b, b]$. We first show that $C^\alpha_{\psi_1}(k)$ and $C^\alpha_{\psi_2}(j)$ are defined in the ranges they are applied in $C^\alpha_\psi(i)$. Let $-x^\tau_j \in -x^\tau_i - I$ and let $-x^\tau_k \in (-x^\tau_j, -x^\tau_i)$ (let $a = b$ if $I = [b, b]$). Then by Definition 6

$$\begin{aligned}
-h + pas(\psi_1) &\leq -h + max(pas(\psi_1), pas(\psi_2)) \\
&= -h + pas(\psi) - b \\
&\leq -x^\tau_i - b \\
&\leq -x^\tau_k \\
&\leq -x^\tau_i \\
&\leq -fut(\psi) \\
&= -max(fut(\psi_1), fut(\psi_2) - a) \\
&\leq -fut(\psi_1)
\end{aligned}$$

and similarly

$$\begin{aligned}
-h + pas(\psi_2) &\leq -h + b + max(pas(\psi_1), pas(\psi_2)) \\
&= -h + pas(\psi) - b \\
&\leq -x_i^\tau - b \\
&\leq -x_j^\tau \\
&\leq -x_i^\tau - a \\
&\leq -fut(\psi) - a \\
&= -max(fut(\psi_1), fut(\psi_2) - a) - a \\
&\leq -fut(\psi_2)
\end{aligned}$$

If $C^0_{\psi_1 \mathcal{S}_{(a,b)} \psi_2}(i)$ is satisfied at t_1, then either $C^0_\top(i)$ is satisfied, and by induction, $\top \in \rho^s_{i'(t_1,i)}$, implying $s, i'(t_1, i) \models \psi$. Otherwise

$$\bigvee_{j=1}^{i-1} \Big[-x_j^\tau \in (-x_i^\tau - b, -x_i^\tau - a) \wedge C^0_{\psi_2}(j) \wedge \bigwedge_{k=j+1}^{i-1} C^0_{\psi_1}(k) \Big]$$

and by induction

$$\bigvee_{j'=l'(t_1)-l+1}^{i'(t_1,i)-1} \Big[\tau^s_{j'} \in (\tau^s_{i'(t_1,i)} - b, \tau^s_{i'(t_1,i)} - a) \wedge s, j' \models \psi_2 \wedge \bigwedge_{k'=j'+1}^{i'(t_1,i)-1} s, k' \models \psi_1 \Big]$$

but then

$$\begin{aligned}
&\exists j' : \tau^s_{j'} \in (\tau^s_{i'(t_1,i)} - b, \tau^s_{i'(t_1,i)} - a).s, j' \models \psi_2 \\
&\wedge \forall k' : j' < k' < i'(t_1, i).s, k' \models \psi_1
\end{aligned}$$

Implying $s, i'(t_1, i) \models \psi$.

Conversely if C^0_ψ is not satisfied at t_1, then $C^0_\top(i)$ is not satisfied, and by induction $\top \notin \rho^s_{i'(t_1,i)}$, and:

$$\bigwedge_{j=1}^{i-1} \Big[-x_j^\tau \in (-x_i^\tau - b, -x_i^\tau - a) \implies \Big(\neg C^0_{\psi_2}(j) \vee \bigvee_{k=j+1}^{i-1} \neg C^0_{\psi_1}(k) \Big) \Big]$$

and by induction

$$\bigwedge_{j'=l'(t_1)-l+1}^{i'(t_1,i)-1} \Big[\tau^s_{j'} \in (\tau^s_{i'(t_1,i)} - b, \tau^s_{i'(t_1,i)} - a) \implies \Big(s, j' \not\models \psi_2 \vee \bigvee_{k'=j'+1}^{i'(t_1,i)-1} s, k' \not\models \psi_1 \Big) \Big]$$

but then

$$\begin{aligned} &\forall j' : \tau^s_{j'} \in (\tau^s_{i'(t_1,i)} - b, \tau^s_{i'(t_1,i)} - a).s, j' \not\models \psi_2 \\ &\vee \exists k' : j' < k' < i'(t_1, i).s, k' \not\models \psi_1 \end{aligned}$$

Implying $s, i'(t_1, i) \not\models \psi$.

The proofs for $C^1_{\psi_1 \mathcal{S}_{(a,b)} \psi_2}(i)$, $C^\alpha_{\psi_1 \mathcal{S}_{[b,b]} \psi_2}(i)$, $C^\alpha_{\psi_1 \mathcal{U}_{(a,b)} \psi_2}(i)$ and $C^\alpha_{\psi_1 \mathcal{U}_{[b,b]} \psi_2}(i)$ are similar.

- $\psi = \psi_1 \mathcal{S}_{(a,\infty)} \psi_2$.

  We first show that $C^\alpha_{\psi_1}(k)$ and $C^\alpha_{\psi_2}(j)$ are defined in all ranges in which they appear in $C^\alpha_\psi(i)$ and in staying conditions and transition guards of D^α_ψ. If $-x^\tau_j \in [-h + pas(\psi) - a - 3, -fut(\psi) - a]$ and $-x^\tau_k \in [-h + pas(\psi) - a - 3, -fut(\psi)]$, then by Definition 6 it holds that

$$\begin{aligned} -h + pas(\psi_1) &\leq -h + max(pas(\psi_1), pas(\psi_2)) \\ &= -h + pas(\psi) - a - 3 \\ &\leq -x^\tau_k \\ &\leq -fut(\psi) \\ &= -max(fut(\psi_1), fut(\psi_2) - a) \\ &\leq -fut(\psi_1) \end{aligned}$$

  and similarly

$$\begin{aligned} -h + pas(\psi_2) &\leq -h + max(pas(\psi_1), pas(\psi_2)) \\ &= -h + pas(\psi) - a - 3 \\ &\leq -x^\tau_j \\ &\leq -fut(\psi) - a \\ &= -max(fut(\psi_1), fut(\psi_2) - a) - a \\ &\leq -fut(\psi_2) \end{aligned}$$

  It is readily seen that this covers all cases by inspection of definitions.

Next we show that D^0_ψ outputs p^0_ψ at t_1 iff the invariant (2) holds at t_1:

If D^0_ψ outputs p^0_ψ at t_1, then it is either taking a transition to state q_2 or is in q_2.

– If D^0_ψ is taking a transition to q_2 at t_1, then the transition guard g^0_2 is satisfied:

$$\bigvee_{j=1}^{l} \Big[-x^\tau_j \in (-h+pas(\psi)-a-3, -h+pas(\psi)-a-2] \wedge C^0_{\psi_2}(j) \\ \wedge \bigwedge_{k=j+1}^{l} \Big(-x^\tau_k \in (-x^\tau_j, -h+pas(\psi)-1] \implies C^0_{\psi_1}(k) \Big) \Big]$$

By induction

$$\bigvee_{j'=l'(t_1)-l+1}^{l'(t_1)} \Big[\tau^s_{j'} \in (t_1-h+pas(\psi)-a-3, t_1-h+pas(\psi)-a-2] \wedge \\ s,j' \models \psi_2 \wedge \bigwedge_{k'=j'+1}^{l'(t_1)} \Big(\tau^s_{k'} \in (\tau^s_{j'}, t_1-h+pas(\psi)-1] \implies s,k' \models \psi_1 \Big) \Big]$$

and thus

$$\exists j' : \tau^s_{j'} \in (t_1-h+pas(\psi)-a-3, t_1-h+pas(\psi)-a-2]. s,j' \models \psi_2 \\ \wedge \forall k' : \tau^s_{k'} \in (\tau^s_{j'}, t_1-h+pas(\psi)-1]. s,k' \models \psi_1$$

Implying (2) holds at t_1.

– If D^0_ψ is in q_2 at t_1, then there is $\delta > 0$ such that D^0_ψ entered q_2 at $t_1 - \delta$ when the transition guard g^0_2 was satisfied and the staying condition I_{q_2} was satisfied for every $t \in (t_1-\delta, t_1]$. By induction:

$$\bigwedge_{k'=l'(t_1-\delta)-l+1}^{l'(t_1)} \Big[\tau^s_{k'} \in (t_1-\delta-h+pas(\psi)-a-2, t_1-h+pas(\psi)-1] \\ \implies s,k' \models \psi_1 \Big]$$

and thus

$$\forall k' : \tau^s_{k'} \in (t_1-\delta-h+pas(\psi)-a-2, t_1-h+pas(\psi)-1]. s,k' \models \psi_1$$

It was previously shown that the invariant (2) held at $t_1 - \delta$. Combining we get:

$$\begin{aligned}\exists j' : &\tau^s_{j'} \in (-\infty, t_1 - h + pas(\psi) - a - 2].s, j' \models \psi_2 \\ \wedge \forall k' : &\tau^s_{k'} \in (\tau^s_{j'}, t_1 - h + pas(\psi) - 1].s, k' \models \psi_1\end{aligned}$$

Implying (2) holds at t_1.

Thus D^0_ψ outputs p^0_ψ at t_1 if (2) holds at t_1.

If D^0_ψ outputs $\neg p^0_\psi$ at t_1, then it is either taking a transition to q_1 or is in q_1:

– If D^0_ψ is taking a transition to q_1 at t_1, then the transition guard g^0_1 is satisfied:

$$\bigvee_{k=1}^{l} \Big[- x^\tau_k \in (-h + pas(\psi) - a - 2, -h + pas(\psi) - 1] \wedge \neg C^0_{\psi_1} \Big]$$

and by induction:

$$\bigvee_{k'=l'(t_1)-l+1}^{l'(t_1)} \Big[\tau^s_{k'} \in (t_1 - h + pas(\psi) - a - 2, t_1 - h + pas(\psi) - 1] \wedge s, k' \not\models \psi_1 \Big]$$

but then

$$\exists k' : \tau^s_{k'} \in (t_1 - h + pas(\psi) - a - 2, t_1 - h + pas(\psi) - 1].s, k' \not\models \psi_1$$

Implying (2) does not hold at t_1.

– If D^0_ψ is in q_1 at time t_1, then there are two possibilities:
  * Either D^0_ψ was in q_1 since the beginning of the run.
  * Or there is $\delta > 0$ such that D^0_ψ entered q_1 using the transition from q_2 at $t_1 - \delta$ when g_1 was satisfied and $I^0_{q_1}$ was satisfied for every $t \in (t_1 - \delta, t_1]$.

  In the first case $I^0_{q_1}$ held from the beginning of the run and thus $\psi_1 \mathcal{S}_{(a,\infty)} \psi_2$ never held, so (2) cannot hold.

  In the second case (2) did not hold at $t_1 - \delta$ and $I^0_{q_1}$ held for every $t \in (t_1 - \delta, t_1]$, and by induction:

$$\begin{aligned}\bigwedge_{j'=l'(t_1)-l+1}^{l'(t_1)} \Big[\tau^s_{j'} \in (t_1 - \delta - h + pas(\psi) - a - 3, t_1 - h + pas(\psi) - a - 2] \implies \\ \Big\{ s, j' \not\models \psi_2 \vee \bigvee_{k'=j'+1}^{l'(t_1)} \Big(\tau^s_{k'} \in (\tau^s_{j'}, t_1 - h + pas(\psi) - 1] \wedge s, k' \not\models \psi_2 \Big) \Big\} \Big]\end{aligned}$$

but then

$$\forall j' : \tau^s_{j'} \in (t_1 - \delta - h + pas(\psi) - a - 3, t_1 - h + pas(\psi) - a - 2].s, j' \not\models \psi_2$$
$$\vee \exists k' : \tau^s_{k'} \in (\tau^s_{j'}, t_1 - h + pas(\psi) - 1].s, k' \not\models \psi_1$$

Combining we have

$$\forall j' : \tau^s_{j'} \in (-\infty, t_1 - h + pas(\psi) - a - 2].s, j' \not\models \psi_2$$
$$\vee \exists k' : \tau^s_{k'} \in (\tau^s_{j'}, t_1 - h + pas(\psi) - 1].s, k' \not\models \psi_1$$

and thus (2) at t_1.

Thus D^0_ψ outputs $\neg p^0_\psi$ at t_1 if (2) does not hold at t_1.
This completes the proof that D^0_ψ outputs p^0_ψ at t_1 iff (2) holds at t_1.
If $C^0_\psi(i)$ is satisfied at t_1, then at least one of the three clauses is satisfied:

- Suppose:
$$C^0_\top(i)$$
By induction $\rho^s_{i'(t_1,i)} = \top$, implying $s, i'(t_1, i) \models \psi$.
- Suppose:
$$p^0_\psi \wedge \bigwedge_{k=1}^{i-1} \Big[-x^\tau_k \in (-h + pas(\psi) - 1, -x^\tau_i) \implies C^0_{\psi_1}(k) \Big]$$
Since p^0_ψ is satisfied (2) holds at t_1, and by induction:
$$\bigwedge_{k=l'(t_1)-l+1}^{i'(t_1,i)-1} \Big[\tau^s_{k'} \in (t_1 - h + pas(\psi) - 1, \tau^s_{i'(t_1,i)}) \implies s, k' \models \psi_1 \Big]$$
and thus
$$\forall k' : \tau^s_{k'} \in (t_1 - h + pas(\psi) - 1, \tau^s_{i'(t_1,i)}).s, k' \models \psi_1$$
Combining with (2) it follows that $s, i'(t_1, i) \models \psi$.
- Suppose:
$$\bigvee_{j=1}^{i-1} \Big[-x^\tau_j \in (-h + pas(\psi) - a - 2, -x^\tau_i - a) \wedge C^0_{\psi_2}(j)$$
$$\wedge \bigwedge_{k=j+1}^{i-1} C^0_{\psi_1}(k) \Big]$$

and by induction

$$\bigvee_{j'=l'(t_1)-l+1}^{i'(t_1,i)-1} \Big[\tau^s_{j'} \in (t_1 - h + pas(\psi) - a - 2, \tau^s_{i'(t_1,i)} - a) \wedge s, j' \models \psi_2 \wedge \bigwedge_{k'=j'+1}^{i'(t_1,i)-1} s, k' \models \psi_1 \Big]$$

and thus

$$\exists j' : \tau^s_{j'} \in (t_1 - h + pas(\psi) - a - 2, \tau^s_{i'(t_1,i)}).s, j' \models \psi_2 \wedge \forall k' : j' < k' < i'(t_1, i).s, k' \models \psi_1$$

Implying $s, i'(t_1, i) \models \psi$

Conversely, if $C^0_\psi(i)$ is not satisfied at t_1, then one of the following clauses is satisfied:

– Suppose:

$$\neg C^0_\top(i) \wedge \neg p^0_\psi \wedge \bigwedge_{j=1}^{i-1} \Big[-x^\tau_j \in [-h + pas(\psi) - a - 2, -x^\tau_i - a) \implies \Big(\neg C^0_{\psi_2}(j) \vee \bigvee_{k=j+1}^{i-1} \neg C^0_{\psi_1}(k) \Big) \Big]$$

and by induction

$$\rho^s_{i'(t_1,i)} \neq \top \wedge \bigwedge_{j'=l'(t_1)-l-1}^{i'(t_1,i)-1} \Big[\tau^s_{j'} \in [t_1 - h + pas(\psi) - a - 2, \tau^s_{i'(t_1,i)} - a) \implies \Big(s, j' \not\models \psi_2 \vee \bigvee_{k'=j'+1}^{i'(t_1,i)-1} s, k' \not\models \psi_1 \Big) \Big]$$

and thus

$$\rho^s_{i'(t_1,i)} \neq \top \wedge \forall j' : \tau^s_{j'} \in [t_1 - h + pas(\psi) - a - 2, \tau^s_{i'(t_1,1)} - a).s, j' \not\models \psi_2 \vee \exists k' : j' < k' < i'(t_1, i).s, k' \not\models \psi_1$$

From $\neg p^0_\psi$ we have that (2) does not hold at t_1. Combining we get:

$$\begin{aligned}
&\rho^s_{i'(t_1,i)} \not\equiv \top \wedge \\
&\quad \forall j' : \tau^s_{j'} \in (-\infty, \tau^s_{i'(t_1,i)} - a).s, j' \not\models \psi_2 \\
&\quad\quad \vee \exists k' : \tau^s_{k'} \in (\tau^s_{j'}, \tau^s_{i'(t_1,i)}).s, k' \not\models \psi_2
\end{aligned}$$

Implying $s, i'(t_1, i) \not\models \psi$.

– Suppose:

$$\begin{aligned}
&\neg C^0_\top(i) \\
&\quad \wedge \bigvee_{k=1}^{i-1} \Big[- x^\tau_k \in (-h + pas(\psi) - 1, -x^\tau_i) \wedge \neg C^0_{\psi_1}(k) \Big] \\
&\quad \wedge \bigwedge_{j=1}^{i-1} \Big[- x^\tau_j \in (-h + pas(\psi) - a - 2, -x^\tau_i - a) \implies \\
&\quad\quad \Big(\neg C^0_{\psi_2}(j) \vee \bigvee_{k=j+1}^{i-1} \neg C^0_{\psi_1}(k) \Big) \Big]
\end{aligned}$$

and by induction:

$$\begin{aligned}
&\rho^s_{i'(t_1,i)} \not\equiv \top \\
&\quad \wedge \bigvee_{k'=l'(t_1)-l+1}^{i'(t_1,i)-1} \Big[\tau^s_{k'} \in (t_1 - h + pas(\psi) - 1, \tau^s_{i'(t_1,i)}) \wedge \neg s, k' \not\models \psi_1 \Big] \\
&\quad \wedge \bigwedge_{j'=l'(t_1)-l+1}^{i'(t_1,i)-1} \Big[\tau^s_{j'} \in (t_1 - h + pas(\psi) - a - 2, \tau^s_{i'(t_1,i)} - a) \implies \\
&\quad\quad \Big(s, j' \not\models \psi_2 \vee \bigvee_{k'=j'+1}^{i'(t_1,i)-1} s, k' \not\models \psi_1 \Big) \Big]
\end{aligned}$$

but then:

$$\begin{aligned}
&\rho^s_{i'(t_1,i)} \not\equiv \top \\
&\quad \wedge \exists k' : \tau^s_{k'} \in (t_1 - h + pas(\psi) - 1, \tau^s_{i'(t_1,i)}).s, k' \not\models \psi_1 \\
&\quad \wedge \forall j' : \tau^s_{j'} \in (t_1 - h + pas(\psi) - a - 2, \tau^s_{i'(t_1,i)} - a). (\ s, j' \not\models \psi_2 \\
&\quad \vee \exists k' : \tau^s_{k'} \in (\tau^s_{j'}, \tau^s_{i'(t_1,i)}).s, k' \not\models \psi_1 \ )
\end{aligned}$$

implying

$$\begin{aligned}\rho^s_{i'(t_1,i)} \neq \top \\ \wedge\forall j' : \tau^s_{j'} \in (-\infty, t_1 - h + pas(\psi) - a - 1).(\ s, j' \not\models \psi_2 \\ \vee\exists k' : \tau^s_{k'} \in (\tau^s_{j'}, \tau^s_{i'(t_1,i)}).s, k' \not\models \psi_1\ ) \\ \wedge\forall j' : \tau^s_{j'} \in (t_1 - h + pas(\psi) - a - 2, \tau^s_{i'(t_1,i)} - a).(\ s, j' \not\models \psi_2 \\ \vee\exists k' : \tau^s_{k'} \in (\tau^s_{j'}, \tau^s_{i'(t_1,i)}).s, k' \not\models \psi_1\ )\end{aligned}$$

Implying $s, i'(t_1, i) \not\models \psi$.

The proof for the case $C^1_\psi(i)$, p^1_ψ and D^1_ψ is similar.

- $\psi = \psi_1 \mathcal{U}_I \psi_2$ where $I = (a, b)$ or $I = [b, b]$. We first show that $C^\alpha_{\psi_1}(k)$ and $C^\alpha_{\psi_2}(j)$ are defined in the ranges they are applied in $C^\alpha_\psi(i)$. Let $-x^\tau_j \in -x^\tau_i + I$ and let $-x^\tau_k \in (-x^\tau_i, -x^\tau_j)$ (let $a = b$ if $I = [b, b]$). Then by Definition 6

$$\begin{aligned}-h + pas(\psi_1) &\leq -h + max(pas(\psi_1), pas(\psi_2) - a) \\ &= -h + pas(\psi) \\ &\leq -x^\tau_i \\ &\leq -x^\tau_k \\ &\leq -x^\tau_i + b \\ &\leq -fut(\psi) + b \\ &= -max(fut(\psi_1), fut(\psi_2)) \\ &\leq -fut(\psi_1)\end{aligned}$$

and similarly

$$\begin{aligned}-h + pas(\psi_2) &\leq -h + max(pas(\psi_1), pas(\psi_2) - a) + a \\ &= -h + pas(\psi) + a \\ &\leq -x^\tau_i + a \\ &\leq -x^\tau_j \\ &\leq -x^\tau_i + b \\ &\leq -fut(\psi) + b \\ &= -max(fut(\psi_1), fut(\psi_2)) \\ &\leq -fut(\psi_2)\end{aligned}$$

The proof for the satisfaction of $C^\alpha_\psi(i)$ is similar to the case for $\psi = \psi_1 \mathcal{S}_{(a,b)} \psi_2$.

- $\psi = \psi_1 \mathcal{U}_{(a,\infty)} \psi_2$. This part of the proof is adapted from [25].

  We first show that $C^\alpha_{\psi_1}(k)$ and $C^\alpha_{\psi_2}(j)$ are defined in all ranges in which they appear in $C^\alpha_\psi(i)$ and in staying conditions and transition guards of D^α_ψ. If $-x^\tau_j \in [-h+pas(\psi)+a, -fut(\psi)+a+1]$ and $-x^\tau_k \in [-h+pas(\psi), -fut(\psi)+a+1]$, then by Definition 6 it holds that

$$\begin{aligned} -h + pas(\psi_1) &\leq -h + max(pas(\psi_1), pas(\psi_2) - a) \\ &= -h + pas(\psi) \\ &\leq -x^\tau_k \\ &\leq -fut(\psi) + a + 1 \\ &= -1 - max(fut(\psi_1), fut(\psi_2)) \\ &\leq -fut(\psi_1) \end{aligned}$$

  and similarly

$$\begin{aligned} -h + pas(\psi_2) &\leq -h + max(pas(\psi_1), pas(\psi_2) - a) + a \\ &= -h + pas(\psi) + a \\ &\leq -x^\tau_j \\ &\leq -fut(\psi) + a + 1 \\ &= -1 - max(fut(\psi_1), fut(\psi_2)) \\ &\leq -fut(\psi_2) \end{aligned}$$

  It is seen that this covers all cases by inspection of definitions.

  Next we show that D^0_ψ outputs p^0_ψ at t_1 iff the invariant (3) holds at t_1:

  If D^0_ψ outputs p^0_ψ at t_1, then it is either in one of the states q_1 or q_2 or is taking a transition to q_1 or q_2:

  - The staying condition $I^0_{q_1}$ of q_1 is

$$\begin{aligned} &\bigwedge_{k=1}^{l} \Big[-x^\tau_k \in (-fut(\psi), -fut(\psi)+a+1) \implies C^0_{\psi_1}(k) \Big] \\ \wedge &\bigwedge_{j=1}^{l} \Big[-x^\tau_j \in (-fut(\psi)+a, -fut(\psi)+a+1) \implies \neg C^0_{\psi_2}(k) \Big] \end{aligned}$$

    If D^0_ψ is in q_1 at t_1, then because q_1 is the only unfair state and the only outgoing transition is to q_2, there is $\delta > 0$ such that g^0_2 is satisfied at $t_1 + \delta$ and $I^0_{q_1}$ is satisfied for every $t \in (t_1, t_1 + \delta)$. By induction:

$$\bigwedge_{k'=l'(t_1)-l+1}^{l'(t_1+\delta)} \Big[\tau^s_{k'} \in (t_1 - fut(\psi), t_1 + \delta - fut(\psi) + a + 1) \implies s, k' \models \psi_1 \Big]$$

and thus

$$\forall k' : \tau^s_{k'} \in (t_1 - fut(\psi), t_1 + \delta - fut(\psi) + a + 1).s, k' \models \psi_1$$

It can easily be seen that g^0_2 corresponds to:

$$\bigvee_{j=1}^{l} \Big[- x^\tau_j \in (-fut(\psi) + a, -fut(\psi) + a + 1] \wedge C^0_{\psi_2}(j) \\ \wedge \bigwedge_{k=1}^{j-1} \Big(- x^\tau_k \in (-fut(-\psi), -x^\tau_j) \implies C^0_{\psi_1}(k) \Big) \Big]$$

and by induction

$$\bigvee_{j'=l'(t_1+\delta)-l+1}^{l'(t_1+\delta)} \Big[\tau^s_{j'} \in (t_1 + \delta - fut(\psi) + a, t_1 + \delta - fut(\psi) + a] \wedge s, j' \models \psi_2 \\ \wedge \bigwedge_{k'=l'(t_1)-l+1}^{j'-1} \Big(\tau^s k' \in (t_1 + \delta - fut(\psi), \tau^s_{j'}) \implies s, k' \models \psi_1 \Big) \Big]$$

but then

$$\exists j' : \tau^s_{j'} \in (t_1 + \delta - fut(\psi) + a, t_1 + \delta - fut(\psi)] \wedge s, j' \models \psi_2 \\ \wedge \forall k' : \tau^s_{k'} \in (t_1 + \delta - fut(\psi), \tau^s_{j'}).s, k' \models \psi_1$$

By combining we get:

$$\exists j' : \tau^s_{j'} \in (t_1 - fut(\psi) + a + \delta, t_1 - fut(\psi) + a + 2 + \delta).s, j' \models \psi_2 \\ \wedge \forall k' \in (t_1 - fut(\psi) + \delta, \tau^s_{k'}).s, k' \models \psi_1$$

Implying (3) holds at t_1.

– The staying condition $I^0_{q_2}$ for q_2 is:

$$\bigvee_{j=1}^{l} \Big[- x^\tau_j \in (-fut(\psi) + a, -fut(\psi) + a + 1) \wedge C^0_{\psi_2}(j) \\ \wedge \bigwedge_{k=1}^{j-1} \Big(- x^\tau_k \in (-fut(\psi), -x^\tau_j) \implies C^0_{\psi_1}(k) \Big) \Big]$$

Thus, if D^0_ψ is in q_2 at t_1, it follows by induction that:

$$\bigvee_{j'=l'(t_1)-l+1}^{l'(t_1)} \Big[\tau^s_{j'} \in (t_1 - fut(\psi) + a, t_1 - fut(\psi) + a + 1) \wedge s, j' \models \psi_2 \\ \wedge \bigwedge_{k'=l'(t_1)-l+1}^{j'-1} \Big(\tau^s_{k'} \in (t_1 - fut(\psi), \tau^s_{j'}) \implies s, k' \models \psi_1 \Big) \Big]$$

and hence

$$\exists j' : \tau^s_{j'} \in (t_1 - fut(\psi) + a, t_1 - fut(\psi) + a + 1).s, j' \models \psi_2$$
$$\wedge \forall k' \in (t_1 - fut(\psi), \tau^s_{j'}).s, k' \models \psi_1$$

Implying (3) holds at t_1.

– If D^0_ψ is taking a transition to q_1 at t_1, then the transition guard g^0_1 is satisfied and thus, $I^0_{q_1}$ holds at t_1. Because the run is accepting, there is $\delta > 0$ such that D^0_ψ stays in q_1 for every $t \in (t_1, t_1 + \delta)$. It was already shown (3) holds for every $t \in (t_1, t_1 + \delta)$. Combining the two, it follows that (3) holds at t_1.

– If D^0_ψ is taking a transition to q_2 at t_1, then the transition guard g^0_2 is satisfied:

$$I^0_{q_2} \vee \left[I^0_{q_1} \wedge \bigvee_{j=1}^{l} \Big(-x^\tau_j = -fut(\psi) + a + 1 \wedge C^0_{\psi_2}(j) \Big) \right]$$

It can easily be seen this corresponds to:

$$\bigvee_{j=1}^{l} \Big[-x^\tau_j \in (-fut(\psi) + a, -fut(\psi) + a + 1] \wedge C^0_{\psi_2}(j)$$
$$\wedge \bigwedge_{k=1}^{j-1} \Big(-x^\tau_k \in (-fut(-\psi), -x^\tau_j) \implies C^0_{\psi_1}(k) \Big) \Big]$$

and by induction:

$$\bigvee_{j'=l'(t_1)-l+1}^{l'(t_1)} \Big[\tau^s_{j'} \in (t_1 - fut(\psi) + a, t_1 - fut(\psi) + a + 1] \wedge s, j' \models \psi_2$$
$$\wedge \bigwedge_{k'=l'(t_1)-l+1}^{j'-1} \Big(\tau^s k' \in (t_1 - fut(\psi), \tau^s_{j'}) \implies s, k' \models \psi_1 \Big) \Big]$$

but then:

$$\exists j' : \tau^s_{j'} \in (t_1 - fut(\psi) + a, t_1 - fut(\psi) + a + 1].s, j' \models \psi_2$$
$$\wedge \forall k' : \tau^s_{k'} \in (t_1 - fut(\psi), \tau^s_{j'}).s, k' \models \psi_1$$

Implying that (3) holds at t_1.

Thus D^0_ψ outputs p^0_ψ at t_1 if (3) holds.

If D^0_ψ outputs $\neg p^0_\psi$ at t_1, then it is either in one of the states q_3 or q_4, or is taking a transition to q_3 or q_4.

– The staying condition $I^0_{q_3}$ of q_3 is:

$$\bigwedge_{k=1}^{l}\Big[-x^\tau_k \in (-fut(\psi), -fut(\psi)+a+1) \implies C^0_{\psi_1}(k)\Big]$$
$$\wedge \bigwedge_{j=1}^{l}\Big[-x^\tau \in (-fut(\psi)+a, -fut(\psi)+a+1) \implies \neg C^0_{\psi_2}(j)\Big]$$

If D^0_ψ is in q_3 at t_1, then because q_3 is a fair state and the only outgoing transition is to q_4, there are two possibilities:

* Either D^0_ψ stays forever in q_3, implying that ψ_2 never becomes true, and thus (3) is not satisfied at t_1.
* Otherwise there is $\delta > 0$ such that g^0_4 is satisfied at $t_1+\delta$ and $I^0_{q_3}$ is satisfied for every $t \in (t_1, t_1+\delta)$. By induction:

$$\bigwedge_{k'=l'(t_1)-l+1}^{l'(t_1+\delta)} \tau^s_{k'} \in (t_1 - fut(\psi), t_1+\delta-fut(\psi)+a+1).s, k' \models \psi_1$$

and thus:

$$\forall k' : \tau^s_{k'} \in (t_1 - fut(\psi), t_1+\delta-fut(\psi)+a+1).s, k' \models \psi_1$$

It can easily be seen that g^0_4 corresponds to:

$$\bigvee_{k=1}^{l}\Big[-x^\tau_k \in (-fut(\psi), -fut(\psi)+a+1] \wedge \neg C^0_{\psi_1}(k) \wedge \bigwedge_{j=1}^{k-1}\Big(-x^\tau_j \in (-fut(\psi)+a, -x^\tau_k) \implies \neg C^0_{\psi_2}(j)\Big)\Big]$$

and by induction:

$$\bigvee_{k'=l'(t_1+\delta)-l+1}^{l'(t_1+\delta)}\Big[\tau^s_{k'} \in (t_1+\delta-fut(\psi), t_1+\delta-fut(\psi)+a+1] \wedge s, k' \not\models \psi_1 \wedge \bigwedge_{j'=l'(t_1+\delta)-l+1}^{k'-1}\Big(\tau^s_{j'} \in (t_1-fut(\psi)+a, \tau^s_{k'}) \implies s, j' \not\models \psi_2\Big)\Big]$$

and thus:

$$\exists k' : \tau^s_{k'} \in (t_1+\delta-fut(\psi), t_1+\delta-fut(\psi)+a+1].s, k' \not\models \psi_1$$
$$\wedge \forall j' : \tau^s_{j'} \in (t_1-fut(\psi)+a, \tau^s_{k'}).s, j' \not\models \psi_2$$

Combining, we get that (3) does not hold at t_1.

– The staying condition $I^0_{q_4}$ for q_4 is

$$\bigvee_{k=1}^{l} \Big[-x^\tau_k \in (-fut(\psi), -fut(\psi)+a+1) \wedge \neg C^0_{\psi_1}(k) \\ \wedge \bigwedge_{j=k+1}^{l} \Big(-x^\tau_j \in (-fut(\psi)+a, -x^\tau_k) \implies \neg C^0_{\psi_2}(j) \Big) \Big]$$

If D^0_ψ is in q_4 at t_1, then $I^0_{q_4}$ is satisfied at t_1, and by induction:

$$\bigvee_{k'=l'(t_1)-l+1}^{l} \Big[\tau^s_{k'} \in (t_1 - fut(\psi), t_1 - fut(\psi)+a+1) \wedge s, k' \not\models \psi_1 \\ \wedge \bigwedge_{j'=k'+1}^{l'(t_1)} \Big(\tau^s_{j'} \in (t_1 - fut(\psi)+a, \tau^s_{k'}) \implies s, j' \not\models \psi_2 \Big) \Big]$$

but then:

$$\exists k' : \tau^s_{k'} \in (t_1 - fut(\psi), t_1 - fut(\psi)+a+1).s, k' \not\models \psi_1 \\ \wedge \forall j' : \tau^s_{j'} \in (t_1 - fut(\psi)+1, \tau^s_{k'}).s, j' \not\models \psi_2$$

Implying (3) does not hold at t_1.

– If D^0_ψ is taking a transition to q_3 at t_1, then the transition guard g^0_3 is satisfied and thus $I^0_{q_3}$ holds at t_1. Because the run is accepting, there is $\delta > 0$ such that D^0_ψ stays in q_3 for every $t \in (t_1, t_1+\delta)$. It was already shown that (3) does not hold for every $t \in (t_1, t_1+\delta)$. Combining the two, it follows that (3) does not hold at t_1.

– If D^0_ψ is taking a transition to q_4 at t_1, then the transition guard g^0_4 is satisfied:

$$\bigvee_{k=1}^{l} \Big[-x^\tau_k \in (-fut(\psi), -fut(\psi)+a+1) \wedge \neg C^0_{\psi_1}(k) \\ \wedge \bigwedge_{j=1}^{k-1} \Big(-x^\tau_j \in (-fut(\psi)+a, -x^\tau_k) \implies C^0_{\psi_2}(j) \Big) \Big]$$

and by induction:

$$\bigvee_{k'=l'(t_1)-l+1}^{l'(t_1)} \Big[\tau^s_{k'} \in (t_1 - fut(\psi), t_1 - fut(\psi)+a+1) \wedge s, k' \not\models \psi_1 \\ \wedge \bigwedge_{j'=l'(t_1)-l+1}^{k'-1} \Big(\tau^s_{j'} \in (t_1 - fut(\psi)+a, \tau^s_{k'}) \implies s, j' \not\models \psi_2 \Big) \Big]$$

but then:

$$\exists k' : \tau^s_{k'} \in (t_1 - fut(\psi), t_1 - fut(\psi) + a + 1).s, k' \not\models \psi_1$$
$$\wedge \forall j' : \tau^s_{j'} \in (t_1 - fut(\psi) + a, \tau^s_{k'}).s, j' \not\models \psi$$

Implying that (3) does not hold at t_1.

Thus D^0_ψ outputs $\neg p^0_\psi$ at t_1 if (3) does not hold.
This completes the proof that D^0_ψ outputs p^0_ψ at t_1 iff (3) holds at t_1.
If $C^0_\psi(i)$ is satisfied at t_1, then one of the following clauses is satisfied:

– Suppose:

$$-x^\tau_i \leq -fut(\psi) \wedge p^0_\psi \wedge \bigwedge_{k=i+1}^{l} \Big[-x^\tau_k \in (-x^\tau_i, -fut(\psi)] \implies C^0_{\psi_1}(k) \Big]$$

By induction:

$$\bigwedge_{k'=i'(t_1)+1}^{l'(t_1)} \Big[\tau^s_{k'} \in (\tau^s_{i'(t_1,i)}, t_1 - fut(\psi)] \implies s, k' \models \psi_1 \Big]$$

and thus:

$$\forall k' : \tau^s_{k'} \in (\tau^s_{i'(t_1,i)}, t_1 - fut(\psi)].s, k' \models \psi_1$$

Because p^0_ψ iff (3) we have:

$$\exists j' : \tau^s_{j'} \in (\tau^s_{i'(t_1,i)} + a, \infty).s, j' \models \psi_2$$
$$\wedge \forall k' \in (\tau^s_{i'(t_1,i)}, \tau^s_{j'}).s, k', \models \psi_1$$

Implying $s, i'(t_1, i) \models \psi$.

– Suppose:

$$-x^\tau_i > -fut(\psi) \wedge \bigvee_{j=i+1}^{l} \Big[-x^\tau_j \in (-x^\tau_i + a, -fut(\psi) + a] \wedge C^0_{\psi_2}(i) \wedge \bigwedge_{k=i+1}^{j-1} C^0_{\psi_1}(k) \Big]$$

By induction:

$$\bigvee_{j'=i'(t_1,i)+1}^{l'(t_1)} \Big[\tau^s_{j'} \in (\tau^s_{i'(t_1,i)} + a, t_1 - fut(\psi) + a] \wedge s, j' \models \psi_2 \wedge \bigwedge_{k'=i'(t_1,i)+1}^{j'-1} s, k' \models \psi_1 \Big]$$

and thus:

$$\begin{aligned}\exists j' :&\tau^s_{j'} \in (\tau^s_{i'(t_1,i)} + a, t_1 - fut(\psi) + a].s, j' \models \psi_2\\ &\wedge \forall k' : i'(t_1, i) < k' < j'.s, k' \models \psi_1\end{aligned}$$

Implying $s, i'(t_1, i) \models \psi$.

Conversely, if $C^0_\psi(i)$ is not satisfied at t_1 then one of the following clauses is satisfied:

- Suppose $-x^\tau_i = -fut(\psi) \wedge \neg p^0_\psi$. Then $\tau^s_{i'(t_1,i)} = t_1 - fut(\psi)$, but then (3) not holding corresponds precisely to $s, i'(t_1, i) \not\models \psi$.
- Suppose $-x^\tau < -fut(\psi)$ and one of:
  - Either:

$$\neg p^0_\psi \wedge \bigwedge_{j=i+1}^{l} \Big[-x^\tau_j \in (-x^\tau_i + a, -fut(\psi) + a] \implies \Big(\neg C^0_{\psi_2}(j) \vee \bigvee_{k=i+1}^{j-1} \neg C^0_{\psi_1}(k) \Big) \Big]$$

    By induction:

$$\bigwedge_{j'=i'(t_1,i)+1}^{l'} \Big[\tau^s_{j'} \in (\tau^s_{i'(t_1,i)} + a, t_1 - fut(\psi) + a] \implies \Big(s, j' \not\models \psi_2 \vee \bigvee_{k'=i'(t_1,i)+1}^{j'-1} s, j' \not\models \psi_1 \Big) \Big]$$

    but then:

$$\begin{aligned}\forall j' : \tau^s_{j'} \in (\tau^s_{i'(t_1,i)} + a, t_1 - fut(\psi) + a].s, j' \not\models \psi_2\\ \vee \exists k' : i'(t_1, i) < k' < j'.s, k' \not\models \psi_1\end{aligned}$$

    From $\neg p^0_\psi$ we have that (3) does not hold at t_1. Combining, we get:

$$\begin{aligned}\forall j' : \tau^s_{j'} \in (\tau^s_{i'(t_1,i)} + a, \infty).s, j' \not\models \psi_2\\ \vee \exists k' : \tau^s_{k'} \in (\tau^s_{i'(t_1,i)}, \tau^s_{j'}).s, k' \not\models \psi_1\end{aligned}$$

    Implying $s, i'(t_1, i) \not\models \psi$

$*$ Or:

$$\bigvee_{k=i+1}^{l} \Big[-x_k^\tau \in (-x_i^\tau, -fut(\psi)] \wedge \neg C^0_{\psi_1}(k) \Big]$$
$$\wedge \bigwedge_{j=i+1}^{l} \Big[-x_j^\tau \in (-x_i^\tau + a, -fut(\psi) + a] \implies \Big(\neg C^0_{\psi_2}(j)$$
$$\vee \bigvee_{k=i+1}^{j-1} \neg C^0_{\psi_1}(k) \Big) \Big]$$

By induction:

$$\bigvee_{k'=i'(t_1,i)+1}^{l'(t_1)} \Big[\tau^s_{k'} \in (\tau^s_{i'(t_1,i)}, t_1 - fut(\psi)] \wedge s, k' \not\models \psi_1 \Big]$$
$$\wedge \bigwedge_{j'=i'(t_1,i)+1}^{l'(t_1)} \Big[\quad \tau^s_{j'} \in (\tau^s_{i'(t_1,i)} + a, t_1 - fut(\psi) + a] \implies \Big(s, j' \not\models \psi_2$$
$$\vee \bigvee_{k'=i'(t_1,i)+1}^{j'-1} \neg s, k' \not\models \psi_1 \Big) \Big]$$

but then:

$$\exists k' : \tau^s_{k'} \in (\tau^s_{i'(t_1,i)}, t_1 - fut(\psi)).s, k' \not\models \psi_1$$
$$\wedge \forall j' : \tau^s_{j'} \in (\tau^s_{i'(t_1,i)} + a, t_1 - fut(\psi) + a].s, j' \not\models \psi_2$$
$$\vee \exists k' : \tau^s_{k'} \in (\tau^s_{i'(t_1,i)}, \tau^s_{j'}).s, k' \not\models \psi_1$$

implying:

$$\forall j' : \tau^s_{j'} \in (t_1 - fut(\psi) + a, \infty).s, j' \not\models \psi_2$$
$$\vee \exists k' : \tau^s_{k'} \in (\tau^s_{i'(t_1,i)}, t_1 - fut(\psi)).s, k' \not\models \psi_1$$
$$\wedge \forall j' : \tau^s_{j'} \in (\tau^s_{i'(t_1,i)} + a, t_1 - fut(\psi) + a].s, j' \not\models \psi_2$$
$$\vee \exists k' : \tau^s_{k'} \in (\tau^s_{i'(t_1,i)}, \tau^s_{j'}).s, k' \not\models \psi_1$$

Implying $s, i'(t_1, i) \not\models \psi$.

The proof for $C^1_\psi(i)$, D^1_ψ and p^1_ψ is similar.

□

Lemma 6. *Assume that for the input word s we have $s, 1 \models \varphi$. Then there is an run ξ of $\mathcal{A}_\varphi$ on w^s which is accepting and in addition it holds that if ξ is in t_1, then for every $i \in \{1, \ldots, l\}$ such that $-x_i^\tau \in [-h + pas(\psi), -fut(\psi)]$ and for every subformula ψ of φ, the constraint $C^0_\psi(t)$ is satisfied at t_1 iff $s, i'(t_1, i) \models \psi$, and the constraint $C^1_\psi(t)$ is satisfied at t_1 iff $\overline{s}, i'(t_1, i) \models \psi$.*

Proof. We show by induction on the structure of the formula φ and subformulas ψ of φ that the required run can ξ be constructed.

- $\psi = p$. The TA for atomic propositions is deterministic and does not affect the acceptance of the run. The proofs that $C^0_p(i)$ is satisfied at t_1 iff $s, i'(t_1, i) \models p$ and $C^1_p(i)$ is satisfied at t_1 iff $\overline{s}, i'(t_1, i) \models p$ are similar to the corresponding cases in the proof of Lemma 5.

- $\psi = \psi_1 \vee \psi_2$, $\psi = \neg\psi$, $\psi = \psi_1 \mathcal{U}_I \psi_2$, $\psi = \psi_1 \mathcal{S}_I \psi_2$ where $I = (a, b)$ or $I = [b, b]$. These formulas do not add states to the automaton. By induction the runs constructed for the automata for ψ_1 and ψ_2 satisfy that $C^0_{\psi_i}(j)$ is satisfied at t_1 iff $s, i'(t_1, j) \models \psi_i$ and $C^1_{\psi_i}(j)$ is satisfied at t_1 iff $\overline{s}, i'(t_1, j) \models \psi_i$.

  The proofs that $C^0_\psi(i)$ is satisfied at t_1 iff $s, i'(t_1, i) \models \psi$ and $C^1_\psi(i)$ is satisfied at t_1 iff $s, i'(t_1, i) \models \psi$ are similar to the the corresponding cases in the proof of Lemma 5.

- $\psi = \psi_1 \mathcal{U}_{(a,\infty)} \psi_2$. This part of the proof is adapted from [25]. We have to show that a run of D^0_ψ can be constructed. Consider the following three conditions:

  1.
  $$\exists j' : \tau^s_{j'} \in (t_1 - fut(\psi) + a, t_1 - fut(\psi) + a + 1).s, j' \models \psi_2$$
  $$\wedge \forall k' : \tau^s_{k'} \in (t_1 - fut(\psi), \tau^s_{j'}).s, k' \models \psi_1$$

  2.
  $$\exists k' : \tau^s_{k'} \in (t_1 - fut(\psi), t_1 - fut(\psi) + a + 1).s, k' \not\models \psi_1$$
  $$\wedge \forall j' : \tau^s_{j'} \in (t_1 - fut(\psi) + a, \tau^s_{k'}).s, j' \not\models \psi_2$$

  3.
  $$\forall k' : \tau^s_{k'} \in (t_1 - fut(\psi), t_1 - fut(\psi) + a + 1).s, k' \models \psi_1$$
  $$\wedge \forall j' : \tau^s_{j'} \in (t_1 - fut(\psi) + a, t_1 - fut(\psi) + a + 1).s, j' \not\models \psi_2$$

  These three conditions are mutually exclusive and their disjunction is valid.

  We partition the timeline, starting at $fut(\psi)$, into a minimum sequence of adjacent intervals $T_1 T_2 \ldots$ such that $T_i = (t_i, t_{i+1})$ with $t_{i+1} > t_i$ and

$t_0 = fut(\psi)$ such that in every interval the input word satisfies one of the three conditions.

We associate to every interval T_i one of the states of D^0_ψ:

- If T_i satisfies (1), then the associated state is q_2.
- If T_i satisfies (2), then the associated state is q_4.
- If T_i satisfies (3), then the associated state is either q_1 or q_3. The non-determinism is resolved by the following three mutually exclusive conditions:
  - If T_{i+1} satisfies the condition (1), then the state associated to T_i is q_1.
  - If T_{i+1} satisfies the condition (2), then the state associated to T_i is q_3.
  - If T_i is the last interval, that is $T_i = (t_i, \infty)$, then the associated state is q_3.

The run of D^0_ψ is generated as a sequence of states according to the above partition of the time line with respect to the input word. The generated run is valid because at every time $t \in T_i$, the run is in a state whose invariant is equal to the condition of T_i, and at every time t_{i+1} between adjacent intervals T_iT_{i+1}, there is a transition between the states in D^0_ψ that are associated with T_i and T_{i+1}, that is:

- Consider a transition entering state q_1. By construction, the only possible transitions are from q_0 and q_2 and q_4.

  It can easily be seen g^0_1 that corresponds to:

$$\bigwedge_{k=1}^{l} \Big[-x^\tau_k \in (-fut(\psi), -fut(\psi) + a + 1] \implies C^0_{\psi_1}(k) \Big] \wedge$$
$$\bigwedge_{j=1}^{l} \Big[-x^\tau_j \in (-fut(\psi) + a, -fut(\psi) + a + 1] \implies \neg C^0_{\psi_2}(j) \Big]$$

  Thus if g^0_1 holds at t_{i+1} it implies $I^0_{q_1}$ holds in some non-singular prefix of T_{i+1}.
- The case for transitions entering q_3 is similar to that for transitions entering q_1.
- Consider a transition entering state q_2. By construction, the only possible transitions are from q_0, q_1 and q_4. It can easily be seen that g^0_2

corresponds to:

$$\bigvee_{j=1}^{l} \Big[-x_j^\tau \in (-fut(\psi)+a, -fut(\psi)+a+1] \wedge C^0_{\psi_2}(j) \\ \wedge \bigwedge_{k=1}^{j-1} \Big(-x_k^\tau \in (-fut(\psi), -x_j^\tau) \implies C^0_{\psi_1}(k) \Big) \Big]$$

Thus if g_2^0 holds at t_{i+1} it implies $I^0_{q_2}$ holds in some non-singular prefix of T_{i+1}.

– Consider a transition entering state q_4. By construction, the only possible transitions are from q_0, q_2 and q_3. It can easily be seen that g_4^0 corresponds to:

$$\bigvee_{k=1}^{l} \Big[-x_k^\tau \in (-fut(\psi), -fut(\psi)+a+1] \wedge \neg C^0_{\psi_1}(k) \\ \wedge \bigwedge_{j=k+1}^{l} \Big(-x_j^\tau \in (-fut(\psi)+a, -x_k^\tau] \implies C^0_{\psi_2}(j) \Big) \Big]$$

Thus if g_4^0 holds at t_{i+1} it implies $I^0_{q_4}$ holds in some non-singular prefix of T_{i+1}.

The generated run is accepting because the only unfair state in D^0_ψ is q_1, and the run can be in q_1 at some $t \in T_i$ only if T_i is followed by T_{i+1} that satisfies condition (1), that is state q_2.

The proof that D^0_ψ outputs p^0_ψ at t_1 iff (3) holds at t_1 is similar to the corresponding case in the proof of Lemma 5. Based on the correctness of p^0_ψ, the proof that at time t_1 the predicate $C^0_\psi(i)$ holds iff $s, i'(t_1, i) \models \psi$ is also similar to the corresponding case in the proof of Lemma 5. The proof for $C^1_\psi(i)$, D^1_ψ and p^1_ψ is similar.

- $\psi = \psi_1 \mathcal{S}_{(a,\infty)} \psi_2$. We have to show that a run of D^0_ψ can be constructed. We partition the timeline into a minimal sequence of adjacent intervals $T_0 T_1 \ldots$ such that $T_i = (t_i, t_{i+1})$ with $t_{i+1} > t_i$ and $t_0 = 0$ such that:

  – In every even numbered interval (T_i such that i is even) the input word satisfies the invariant (2) and the associated state is q_1.

  – In every odd numbered interval (T_i such that i is odd) the input word does not satisfy (2) and the associated state is q_2.

The run of D^0_ψ is generated as a sequence of states according to the above partition of the timeline with respect to the input word. The generated run is valid because at every time $t \in T_i$ the run is in a state associated with T_i, and at every time t_{i+1} between two adjacent intervals T_iT_{i+1}, there is a transition between the states in D^0_ψ that are associated with T_i and T_{i+1}

We now demonstrate that the run generated in this way is valid:

- The state associated with T_0 is q_1, and this agrees with the structure of D^0_ψ which has q_1 as initial state. The staying condition $I^0_{q_1}$ for q_1 is given by:

$$\bigwedge_{j=1}^{l} \Big[-x^\tau_j \in (-h + pas(\psi) - a - 3, -h + pas(\psi) - a - 2] \implies \Big\{ \neg C^0_{\psi_2}(j) \vee \bigvee_{k=j+1}^{l} \Big(-x^\tau_k \in (-x^\tau_j, -h + pas(\psi) - 1] \wedge \neg C^0_{\psi_1}(k) \Big) \Big\} \Big]$$

  We need to show that there is $\delta > 0$ such that $I^0_{q_1}$ holds for every time $t \in [0, \delta)$. It is easy to see that $I^0_{q_1}$ holds vacuously as long as no clock variable $-x^\tau_j$ is in the interval specified by the outermost clause in $I^0_{q_1}$. By inspecting Definition 6 it can be seen that $h = fut(\varphi) + pas(\varphi) \geq pas(\psi)$ and as $a \geq 0$ it follows that $-h + pas(\psi) - a - 2 < 0$. Because at time $t_0 = 0$ every clock x^τ_j is 0 or $\perp$ it follows that no clock value $-x^\tau_j$ enters the interval in the outermost clause of $I^0_{q_1}$ until earliest time $\delta = h - pas(\psi) + a + 2$.

- The only possible transition to q_1 is from q_2, and because the transition guard g^0_1 is the negation of the staying condition $I^0_{q_2}$ for q_2, it is always possible to continue a run from q_2 by transitioning to q_1. If g^0_1 is satisfied at t_{i+1}, then:

$$\bigvee_{k=1}^{l} \Big[-x^\tau_k \in (-h + pas(\psi) - a - 2, -h + pas(\psi) - 1] \wedge \neg C^0_{\psi_1}(k) \Big]$$

  To see that there is $\delta > 0$ such that g^0_1 holds for every $t \in [t_{i+1}, t_{i+1} + \delta)$, let $k \in \{1, \ldots, l\}$ be the the smallest index such that $-x^\tau_k \in (-h + pas(\psi) - a - 2, -h + pas(\psi) - 1]$ and $\neg C^0_{\psi_1}(k)$ is satisfied at t_{i+1} and let $\delta = h - pas(\psi) + a + 2 - x^\tau_k$.

  Further, $t \in [t_{i+1}, t_{i+1} + \delta)$ and for every $j \in \{1, \ldots, l\}$ such that $-x^\tau_j \in (-h + pas(\psi) - a - 3, -h + pas(\psi) - a - 1]$ has $-x^\tau_k \in (-x^\tau_j, -h +$

$pas(\psi) - 1]$, implying $I^0_{q_1}$ holds:

$$\bigwedge_{j=1}^{l} -x^\tau_j \in (-h + pas(\psi) - a - 3, -h + pas(\psi) - a - 2] \implies \neg C^0_{\psi_2}(j)$$
$$\vee \bigvee_{k=j+1}^{l} -x^\tau_k \in (-x^\tau_j, -h + pas(\psi) - 1] \wedge \neg C^0_{\psi_1}(k)$$

– The only possible transition to q_2 is from q_1, and because the transition guard g^0_2 is the negation of the staying condition $I^0_{q_1}$ for q_1, is is always possible to continue a run from q_1 by transitioning to q_2. If g^0_2 is satisfied at t_{i+1}, then:

$$\bigvee_{j=1}^{l} \Big[-x^\tau_j \in (-h + pas(\psi) - a - 3, -h + pas(\psi) - a - 2] \wedge C^0_{\psi_2}(j)$$
$$\wedge \bigwedge_{k=j+1}^{l} \Big(-x^\tau_k \in (-x^\tau_j, -h + pas(\psi) - 1] \implies C^0_{\psi_1}(k) \Big) \Big]$$

It follows that there is $j \in \{1, \ldots, l\}$ such that $-x^\tau_j \in (-h + pas(\psi) - a - 3, -h + pas(\psi) - a - 2]$ and $C^0_{\psi_1}(j)$ and $\bigwedge_{k=j+1}^{l} -x^\tau_k \in (-x^\tau_j, -h + pas(\psi) - 1] \implies C^0_{\psi_1}(k)$ are satisfied at t_{1+1}, but then because $(-h + pas(\psi) - a - 2, -h + pas(\psi) - 1] \subseteq (-x^\tau_j, -h + pas(\psi) - 1]$, it follows that $I^0_{q_2}$ is satisfied at t_{i+1}:

$$\bigwedge_{k=1}^{l} \Big[-x^\tau_k \in (-h + pas(\psi) - a - 2, -h + pas(\psi) - 1] \implies C^0_{\psi_1}(k) \Big]$$

Note that $I^0_{q_2}$ holds vacuously if there is no k such that $-x^\tau_k$ lies in the specified interval. Let $k_{min} \in \{1, \ldots, l\}$ be the smallest index such that $-x^\tau_{k_{min}} \in (-h + pas(\psi) - a - 2, -h + pas(\psi) - 1]$. If $k_{min} = l$ let $\delta = h - pas(\psi) + 1$, otherwise $\delta = h - pas(\psi) + 1 - x^\tau_{k_{min}+1}$. Then $I^0_{q_2}$ holds for every $t \in [t_{i+1}, t_{i+1} + \delta)$.

We have shown there is $T_0 = [0, t_2)$ where $I^0_{q_1}$ holds throughout.Because g^0_2 is the negation of $I^0_{q_1}$ it is possible to transition from q_1 to q_2 at $t_{2 \cdot i+1}$, and in a similar way g^0_1 is the negation of $I^0_{q_2}$ and it is possible to transition from q_2 to q_1 at $t_{2 \cdot i}$.

All states of D^0_ψ are fair and thus all runs are accepting.

The proof that D^0_ψ outputs p^0_ψ at t_1 iff (2) holds at t_1 is similar to the corresponding case in the proof of Lemma 5. Based on the correctness of p^0_ψ, the proof that at time t_1 the constraint $C^0_\psi(i)$ holds iff $s, i'(t_1, i) \models \psi$ is also similar to the corresponding case in the proof of Lemma 5. The proof for $C^1_\psi(i)$, D^1_ψ and p^1_ψ is similar.

□

From Lemma 5 $w^s \in \mathcal{L}(A_\varphi)$ implies $s, 1 \models \varphi$, and from Lemma 6 $s, 1 \models \varphi$ implies $w^s \in \mathcal{L}(\mathcal{A}_\varphi)$. It follows that:

Theorem 1. *For timed word $s = (\rho^s, \tau^s)$ and formula φ we have $s, 1 \models \varphi$ iff $w^s \in \mathcal{L}(\mathcal{A}_\varphi)$.*

Corollary 1. *For every MTL formula φ with m propositions, $n_\mathcal{U}$ unbounded until operators, $n_\mathcal{S}$ unbounded since operators, inputs of bounded variability k_{var} and event rate k_{ER}, there exists a nondeterministic timed automaton $\mathcal{A}_\varphi$ with $h \cdot \left(2 \cdot \lceil \frac{k_{var}}{2} \rceil + k_{ER}\right)$ clocks and $(2^m + 1) \cdot (2 \cdot 4^{n_\mathcal{U}} + 1) \cdot 2^{n_\mathcal{S}}$ states that accepts every instance s of the language of φ packaged as a signal w^s.*

Notice that this improves the number of states of $\mathcal{A}_\varphi$ compared to [25] when φ contains no since operators from $\left(\left(2 \cdot \lceil \frac{k_{var} \cdot fut(\varphi)}{2} \rceil + 1\right)^m + 1\right) \cdot (2 \cdot 4^{n_\mathcal{U}} + 1)$ states to $(2^m + 1) \cdot (2 \cdot 4^{n_\mathcal{U}} + 1)$ states.

Our construction thus captures the target semantics, which was the goal.

4.8 Monitoring

This subsection outlines how the construction of $\mathcal{A}_\varphi$ can be used to obtain a monitoring procedure. Correct behavior of a program is specified as an MTL formula φ and a trace from the program is represented by a timed word s.

We construct two TAs $\mathcal{A}_\varphi = \mathcal{A}_{MEM} \otimes D_\varphi$ for φ and $\mathcal{A}_{\neg\varphi} = \mathcal{A}_{MEM} \otimes D_\varphi$ for its negation $\neg\varphi$ respectively. By Definition 1, Definition 2, Theorem 1 and Lemma 4 we have:

- $w^{s^\top} \in \mathcal{L}(\mathcal{A}_\varphi)$ iff $s, 1 \models^- \varphi$, and
- $w^{s^\top} \notin \mathcal{L}(\mathcal{A}_{\neg\varphi})$ iff $s, 1 \models^+ \varphi$.

Rather than determining the above directly by evaluating $\mathcal{A}_\varphi$ and $\mathcal{A}_{\neg\varphi}$ on the infinite signal $w^{s^\top}$, which could in general require an infinite number of steps, we use more indirect but finite means. First evaluate $\mathcal{A}_\varphi$ and $\mathcal{A}_{\neg\varphi}$ on w^s and track the states Q^s_φ of D_φ reached after all of w^s has been consumed and denote the

corresponding states of $D^s_{\neg\varphi}$ by $Q^s_{\neg\varphi}$. Denote the clock valuations of $\mathcal{A}_{MEM}$ after consuming all of w^s by v^s.

Let $\mathcal{A}_{GEN}$ be a TA which has the same number of clocks as $\mathcal{A}_{MEM}$ and which generates the timed word $\top^\omega_{\varphi,s}$ in the sense that regardless of what input signal is given to $\mathcal{A}_{GEN}$, it makes clock assignments identical to those made by $\mathcal{A}_{MEM}$ when given the timed word $\top^\omega_{\varphi,s}$ packaged as a signal $w^{\top^\omega_{\varphi,s}}$. Given that $s^\top$ is of bounded event rate k_{ER} such a TA can easily be constructed.

Let $\mathcal{A}'_\varphi = \mathcal{A}_{GEN} \otimes D_\varphi$ and in addition let v^s be its initial clock valuation and let Q^s be its set of initial states. In the same way define $\mathcal{A}'_{\neg\varphi} = \mathcal{A}_{GEN} \otimes D_{\neg\varphi}$ with initial clock valuation v^s and set of initial states $Q^s_{\neg\varphi}$. With these definitions we have:

- $w^{s^\top} \in \mathcal{L}(\mathcal{A}_\varphi)$ iff $\mathcal{A}'_\varphi$ accepts every infinite signal, and
- $w^{s^\top} \notin \mathcal{L}(\mathcal{A}_{\neg\varphi})$ iff $\mathcal{A}'_{\neg\varphi}$ accepts no signal.

In both instances the problem is reduced to checking the corresponding TA for emptiness. $\mathcal{A}'_\varphi$ (and correspondingly $\mathcal{A}'_{\neg\varphi}$) is non-empty iff its *region graph* [1] has representatives from each of its sets of accepting states belonging to non-trivial strongly connected components reachable from some state in Q^s_φ (and correspondingly $Q^s_{\neg\varphi}$). This can be decided by depth-first searches starting from each state in Q^s_φ ($Q^s_{\neg\varphi}$) of the region graph of $\mathcal{A}'_\varphi$ ($\mathcal{A}'_{\neg\varphi}$).

We have thus demonstrated how our construction can be used for monitoring, which was our goal.

Chapter 5

Related work

5.1 ω-automata and timed automata

Standard finite automata [19] are used to represent formal languages which are sets of finite words over finite alphabets. By modifying their acceptance conditions in a suitable way, similar constructions called ω-automata, can be used to represent infinite words over finite alphabets. An accessible introduction to ω-automata is given in [30]. Standard algorithms for automata provide set operations on formal languages including union, intersection and complementation as well as emptiness checking which is the problem of determining whether a language has at least one member or is the empty set.

Timed automata [1, 2] were introduced to model the behavior of real-time systems over time. A timed automaton can be turned into an equivalent ω-automaton referred to as its *region* automaton and emptiness can be checked in time exponential in the number of clocks and the length of the clock constraints of the timed automaton. Bengtsson and Yi [10] cover decision problems and algorithms for timed automata.

5.2 LTL, MTL and MITL

LTL was introduced by Pnueli [27] and MTL was introduced by Koymans [20]. MITL was introduced as a relaxation of MTL by Alur et al. [3]. The standard construction for translating LTL into ω-automata is given in [31] and explained in [30].

5.2.1 Translation MTL and MITL to timed automata

Maler et al. [23], Ferrère et al. [16] give a translation from continuous time MITL to timed automata based on *temporal testers*. Every subformula is built as a timed automaton which reads the truth value of its subformulas and itself outputs its truth value. Due to the relaxation of MITL compared to MTL they are able to handle signals of arbitrary variability using a finite number of clocks. A drawback to their approach is that clock assignments are mixed with non-determinism, resulting in timed automata that cannot be determinized.

Ničković and Piterman [25] give a translation from continuous-time MTL to timed automata. Crucially they separate clock assignments from non-determinism imposed by unbounded future operators and this makes the construction determinizable. The translation form MTL to timed automata given in this thesis is based on Ničković and Piterman [25] but our approach is for point-in-time MTL over finite and infinite traces. In addition [25] requires quantifier elimination for its clock constraints and in the worst case this increases the size of the predicates by an exponential factor. We are able to avoid this thanks to working in a different time domain. Our construction supports both the past and future fragments of MTL while [25] only supports the future fragment.

5.3 Runtime verification

Runtime verification has been surveyed in Leucker and Schallhart [22] and more recently in Bartocci et al. [4]. Sánchez et al. [28] is a recent survey of open problems and challenges in runtime verification.

5.3.1 Finite traces and monitorability

LTL semantics is defined over infinite words but traces from running programs are necessarily finite. Various semantics for LTL over finite traces have been proposed with *three-valued semantics* by Bauer et al. [6] and *truncated path semantics* by [14] having received the most attention in the literature. Both of these semantics are related to the notion of *good and bad prefixes* in [21]. A finite word x over Σ is a good prefix of language $\mathcal{L}(\varphi)$ iff for all y over Σ the concatenation $x \cdot y$ is in $\mathcal{L}(\varphi)$, and x is a bad prefix of language $\mathcal{L}(\varphi)$ iff for all words y over Σ the concatenation $x \cdot y$ is not in $\mathcal{L}(\varphi)$. A distinguished subcategory of bad prefixes are *informative prefixes* that "tell the whole story" of why they are not in $\mathcal{L}(\varphi)$.

In three-valued semantics the verdict given for word x on formula φ is $\bot$ if x is a bad prefix of φ, $\top$ if x is a good prefix of φ and ? if x is neither. Truncated path semantics give verdicts according to *weak* and *strong* views where weak rejection

for x on φ corresponds to x being an informative prefix of φ and strong acceptance corresponds to x being an informative prefix of $\neg\varphi$.

Monitorability which is concerned with classifying properties into categories based on what guarantees can be given by only observing finite traces has been studied by Bauer et al. [7] and Peled and Havelund [26].

5.3.2 LTL and MTL runtime monitoring

Numerous runtime monitoring systems have been described in the literature. Here we briefly survey some representative samples for monitoring of LTL and MTL specifications.

Havelund and Rosu [17] suggest a procedure based on dynamic programming for online monitoring of the past fragment of LTL.

Thati and Rosu [29] give a procedure based on dynamic programming for point-in-time MTL over finite traces with discrete timestamps. The semantics they implement are non-standard.

Bauer et al. [6, 8] give a procedure for online monitoring of LTL and a timed variation of LTL, both of them with three-valued semantics. Their procedure is based on non-deterministic automata where every state q is annotated with the verdict for the input word x if q is the final state when reading x.

Ho et al. [18] give an online monitoring procedure for point-in-time MTL with truncated path semantics based on a combination of dynamic programming and two-way alternating automata. Their procedure is *trace-length independent* [9], meaning its memory usage is independent of the length of the trace. A weakness of their work is that they rely on rewriting the unbounded parts of the formulas into LTL, which in worst case has non-elementary blowup of the size of the formula.

Basin et al. [5] suggest *event-rate independence*, meaning that memory usage should not depend on the event rate of the trace, and the procedure they present has memory usage logarithmic in the event rate of the trace. Their procedure is based on dynamic programming. From the future fragment they only support bounded future operators.

Chapter 6

Conclusion

This thesis started out with the observation that translations from MTL to timed automata have been used surprisingly little in runtime verification despite the strong connections between temporal logics and automata theory and the fact that timed automata themselves are defined over timed words and signals just like MTL.

We then proposed to begin filling this gap by adapting an existing translation from MTL to timed automata for use in runtime monitoring, and we picked Ničković and Piterman [25] for this. Several requirements related to semantics for finite traces and optimizations that such a translation must satisfy where then specified. These were summarized as research hypothesis **H1**. Theorem 1 establishes the equivalence between the resulting adapted translation and the semantics specified in Chapter 2, and this confirms **H1**.

The second part of our research objectives and its resulting hypothesis **H2** was about demonstrating that the resulting translation from the given MTL semantics to timed automata can provide a feasible procedure for runtime monitoring. In Section 4.8 we demonstrated that runtime monitoring can be done using our translation. We thus partly confirmed **H2** in the sense that the translation can be the basis of a monitoring procedure, but as we have not implemented the procedure we are not able to assess its feasibility for practical use.

6.1 Future work

Implementation of the monitoring procedure can be achieved by translating timed automata to their *region automata* [2] and the algorithms covered by Bengtsson and Yi [10] may be used for that. We also believe further investigation into the structure of the region automata resulting from our translation from MTL would enable further optimizations and improvements directly relevant to monitoring.

The monitoring procedure given in Section 4.8 does offline monitoring in that it reads a full trace and gives a single verdict for the entirety of the trace. In contrast an, online monitor is given events one at a time and gives a verdict for the prefix observed thus far is more useful. For example, if online monitoring is integrated into a system, the system may be able to directly take action when a subsystem fails to follow its specification. One way of obtaining an online monitoring procedure from our translation is by on-the-fly computing and memorizing reachability to accepting cycles from every state of the region automaton, in way similar to what is done in Bauer et al. [8].

We also believe further optimizations of the timed automata are possible. Checking a timed automaton $\mathcal{A}$ for emptiness by translating it to region automata has running time $O\left(|Q| \cdot 2^{|\delta(\mathcal{A}|}\right)$, where $|Q|$ is the number of states of $\mathcal{A}$ and $\delta(\mathcal{A})$ is the size of the clock constraints of $\mathcal{A}$. By Corollary 1 the timed automaton resulting from our translation for a formula φ has $|Q|$ exponential in the number of unbounded until operators of the φ. This suggests a possible trade-off between the number of states and the size of the clock constraints. For example one could change the timed automata for memorizing atomic proposition from Section 4.3.2 to encode the truth values of propositions using states rather than using clocks to memorize the intervals during which the propositions are true. Having done that the remaining clock constraints are due to the clocks of the timed automaton for memorization of punctual events (Section 4.3.1), and all of these clocks satisfy $x_i^\tau < x_{i+}^\tau$ for every $1 \leq i < l$, which may be exploited to reduce the number of states of the region automaton.

Chapter 7

Bibliography

[1] Rajeev Alur. Timed automata. In *Computer Aided Verification, 11th International Conference, CAV '99, Trento, Italy, July 6-10, 1999, Proceedings*, pages 8–22, 1999.

[2] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.

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

[4] Ezio Bartocci, Yliès Falcone, Adrian Francalanza, and Giles Reger. Introduction to runtime verification. In Ezio Bartocci and Yliès Falcone, editors, *Lectures on Runtime Verification - Introductory and Advanced Topics*, volume 10457 of *Lecture Notes in Computer Science*, pages 1–33. Springer, 2018.

[5] David A. Basin, Bhargav Nagaraja Bhatt, and Dmitriy Traytel. Almost event-rate independent monitoring of metric temporal logic. In Axel Legay and Tiziana Margaria, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 23rd International Conference, TACAS 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings, Part II*, volume 10206 of *Lecture Notes in Computer Science*, pages 94–112, 2017.

[6] Andreas Bauer, Martin Leucker, and Christian Schallhart. Monitoring of real-time properties. In S. Arun-Kumar and Naveen Garg, editors, *FSTTCS 2006: Foundations of Software Technology and Theoretical Computer Science, 26th International Conference, Kolkata, India, December 13-15, 2006, Proceedings*, volume 4337 of *Lecture Notes in Computer Science*, pages 260–272. Springer, 2006.

[7] Andreas Bauer, Martin Leucker, and Christian Schallhart. Comparing LTL semantics for runtime verification. *J. Log. Comput.*, 20(3):651–674, 2010.

[8] Andreas Bauer, Martin Leucker, and Christian Schallhart. Runtime verification for LTL and TLTL. *ACM Trans. Softw. Eng. Methodol.*, 20(4):14:1–14:64, 2011.

[9] Andreas Bauer, Jan-Christoph Küster, and Gil Vegliach. From propositional to first-order monitoring. In Axel Legay and Saddek Bensalem, editors, *Runtime Verification - 4th International Conference, RV 2013, Rennes, France, September 24-27, 2013. Proceedings*, volume 8174 of *Lecture Notes in Computer Science*, pages 59–75. Springer, 2013.

[10] Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In Jörg Desel, Wolfgang Reisig, and Grzegorz Rozenberg, editors, *Lectures on Concurrency and Petri Nets, Advances in Petri Nets [This tutorial volume originates from the 4th Advanced Course on Petri Nets, ACPN 2003, held in Eichstätt, Germany in September 2003. In addition to lectures given at ACPN 2003, additional chapters have been commissioned]*, volume 3098 of *Lecture Notes in Computer Science*, pages 87–124. Springer, 2003.

[11] Yves Bertot and Pierre Castéran. *Interactive Theorem Proving and Program Development - Coq'Art: The Calculus of Inductive Constructions*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2004.

[12] Doron Bustan, Dana Fisman, and John Havlicek. Automata construction for psl. Technical report, The Weizmann Institute of Science, 2005.

[13] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model checking*. MIT Press, 2001.

[14] Cindy Eisner, Dana Fisman, John Havlicek, Yoad Lustig, Anthony McIsaac, and David Van Campenhout. Reasoning with temporal logic on truncated paths. In *International conference on computer aided verification*, pages 27–39. Springer, 2003.

[15] Cindy Eisner, Dana Fisman, John Havlicek, and Johan Mårtensson. The, $\top$, $\bot$ approach for truncated semantics. Technical report, Technical report, Accellera, 2006.

[16] Thomas Ferrère, Oded Maler, Dejan Nickovic, and Amir Pnueli. From real-time logic to timed automata. *J. ACM*, 66(3):19:1–19:31, 2019.

[17] Klaus Havelund and Grigore Rosu. Monitoring programs using rewriting. In *16th IEEE International Conference on Automated Software Engineering (ASE 2001), 26-29 November 2001, Coronado Island, San Diego, CA, USA*, pages 135–143. IEEE Computer Society, 2001.

[18] Hsi-Ming Ho, Joël Ouaknine, and James Worrell. Online monitoring of metric temporal logic. In Borzoo Bonakdarpour and Scott A. Smolka, editors, *Runtime Verification - 5th International Conference, RV 2014, Toronto, ON, Canada, September 22-25, 2014. Proceedings*, volume 8734 of *Lecture Notes in Computer Science*, pages 178–192. Springer, 2014.

[19] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

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

[21] Orna Kupferman and Moshe Y. Vardi. Model checking of safety properties. In Nicolas Halbwachs and Doron A. Peled, editors, *Computer Aided Verification, 11th International Conference, CAV '99, Trento, Italy, July 6-10, 1999, Proceedings*, volume 1633 of *Lecture Notes in Computer Science*, pages 172–183. Springer, 1999.

[22] Martin Leucker and Christian Schallhart. A brief account of runtime verification. *J. Log. Algebraic Methods Program.*, 78(5):293–303, 2009.

[23] Oded Maler, Dejan Nickovic, and Amir Pnueli. From MITL to timed automata. In Eugene Asarin and Patricia Bouyer, editors, *Formal Modeling and Analysis of Timed Systems, 4th International Conference, FORMATS 2006, Paris, France, September 25-27, 2006, Proceedings*, volume 4202 of *Lecture Notes in Computer Science*, pages 274–289. Springer, 2006.

[24] Glenford J. Myers. *The art of software testing (2. ed.).* Wiley, 2004.

[25] Dejan Ničković and Nir Piterman. From MTL to deterministic timed automata. In *International Conference on Formal Modeling and Analysis of Timed Systems*, pages 152–167. Springer, 2010.

[26] Doron Peled and Klaus Havelund. Refining the safety-liveness classification of temporal properties according to monitorability. In Tiziana Margaria, Susanne Graf, and Kim G. Larsen, editors, *Models, Mindsets, Meta: The What, the How, and the Why Not? - Essays Dedicated to Bernhard Steffen on the Occasion of His 60th Birthday*, volume 11200 of *Lecture Notes in Computer Science*, pages 218–234. Springer, 2018.

[27] Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, pages 46–57. IEEE Computer Society, 1977.

[28] César Sánchez, Gerardo Schneider, Wolfgang Ahrendt, Ezio Bartocci, Domenico Bianculli, Christian Colombo, Yliès Falcone, Adrian Francalanza, Srdan Krstic, João M. Lourenço, Dejan Nickovic, Gordon J. Pace, José Rufino, Julien Signoles, Dmitriy Traytel, and Alexander Weiss. A survey of challenges for runtime verification from advanced application domains (beyond software). *Formal Methods Syst. Des.*, 54(3):279–335, 2019.

[29] Prasanna Thati and Grigore Rosu. Monitoring algorithms for metric temporal logic specifications. *Electron. Notes Theor. Comput. Sci.*, 113:145–162, 2005.

[30] Moshe Y. Vardi. An automata-theoretic approach to linear temporal logic. In Faron Moller and Graham M. Birtwistle, editors, *Logics for Concurrency - Structure versus Automata (8th Banff Higher Order Workshop, Banff, Canada, August 27 - September 3, 1995, Proceedings)*, volume 1043 of *Lecture Notes in Computer Science*, pages 238–266. Springer, 1995.

[31] Moshe Y. Vardi and Pierre Wolper. An automata-theoretic approach to automatic program verification (preliminary report). In *Proceedings of the Symposium on Logic in Computer Science (LICS '86), Cambridge, Massachusetts, USA, June 16-18, 1986*, pages 332–344. IEEE Computer Society, 1986.