Specifying Safety-Critical Systems with a Decidable Interval Temporal Logic

Savas Konur

Department of Computer Science, University of Liverpool
Liverpool, L69 3BX, UK
konur@liverpool.ac.uk

Abstract

Punctual timing constraints are important in formal modeling of safety-critical real-time systems. But they are very expensive to express in dense time. In most cases, punctuality and dense-time lead to undecidability. Efforts have been successful to obtain decidability; but the results are either non-primitive recursive or nonelementary. In this paper we propose an interval temporal logic which can express quantitative temporal constraints and *punctuality timing constraints* over *continuous* intervals and has a reasonable complexity. Our logic allows most specifications that are interesting in practice, and retains punctuality. It can capture the semantics of both *events* and *states*, and incorporates the notions *duration* and *accumulation*. We call this logic *ESDL* (the acronym stands for *Event- and State-based Duration Logic*). We show that the satisfiability problem is *decidable*, and the complexity of the satisfiability problem is *NEXPTIME*. ESDL is one of a few decidable interval temporal logics with metric operators. Through some case studies, we also show that ESDL can specify many safety-critical real-time system properties which were previously specified by undecidable interval logics or their decidable reductions based on some abstractions.

Keywords: safety-critical systems, punctuality timing constraints, temporal logics, decidability, tableau systems

1. Introduction

Formal modeling of real-time requirements is crucial in analysing safety-critical systems. Among these, *punctuality*[1] properties constitute an important part of the real-time requirements. There have been various modeling paradigms to make this analysis mathematically possible. Two main approaches for modeling time are *dense/continous* modeling of time and *discrete* time models [1].

[1] A *punctuality property* states, for example, that an event/state B follows an event/state A in exactly 3 seconds.

Discrete time points provide an advantage in terms of obtaining decidability and reasonable complexity; but they are not natural ways of modeling real-time systems. On the other hand, dense/continuous time domains mainly lead to undecidability for punctuality properties [2]. Indeed, until recently researchers thought that any linear-time temporal logic which can express punctuality properties is undecidable [3]. There have been recent attempts to obtain decidability; but the results are either non-primitive recursive or nonelementary [4, 5, 3]. In a recent study an EXPSPACE complexity is found [3] for a fragment of ML; but the resulting logic has some restrictions.

These reported results are important, but they are not sufficient because complexities of these logics are still very high to use them in the formal analysis of real-time systems. An alternative approach is to relax density/continuity, and use feasible time models (e.g. discrete); but in this approach we cannot capture exact system behaviour. This is an important limitation in real-time system study. In particular, safety-cricital systems should be designed and analysed according to exact specifications; but we cannot achieve this using simple time models. Therefore, a realistic (i.e. dense or continuous) modeling of time should be used.

One aim of this paper is to tackle these issues. We propose a logic, called *ESDL* (the acronym stands for *Event- and State-based Duration Logic*), which can express punctuality timing constraints over continuous-time intervals. The proposed logic is decidable, and has a reasonable complexity.

Another design choice is to choose between *points* (instants) or *intervals* for modeling time. The interval-based scheme provides a richer representation formalism than the point-based scheme to specify continuous processes and temporal statements; for example, intervals can support uncertain temporal relations between real world events, which the point-base scheme cannot support [6].

Quite to the contrary, interval temporal logics, such as , e.g. ITL [7], RTIL [8], TILCO [9], DC [10], NL [11], PNL [12], are more likely to be undecidable. In the literature, various methods have been proposed to achieve decidability for interval logics. However, most of these methods, such as translating interval logics into discrete or sampled time, cause some syntactic and semantic restrictions. For this reason, we use intervals as primitive objects of time models. Unlike many other interval logics, we do not apply any abstraction to time models, and we therefore try to minimise semantic restrictions. Yet, we can still acquire decidability and a reasonable complexity.

One important design paradigm is to choose between *events* and *states*[2]. A formal method should cover both event-based and state-based views in order to model the behaviour of real-time systems more accurately. In early phases

[2]Consider the sentences "John wrote the letter", and "John worked on the letter". The former reports an event, while the latter reports an ongoing process or state-types. The sentences which describe state-types have the property that whenever they are true over an interval I, they are true over every subinterval of I. Other sentences describing completed events have the property that whenever they are true over an interval I, they are false over some subinterval of I.

of system development, requirements are usually provided less formally; it is therefore more convenient to specify them with event-based formalisms; on the other hand, in later and implementation stages, state-based methods are more convenient, because programming languages are generally state-based [13]. Most temporal logics employed for the specification of real-time systems, however, do not support both methods. They are either event-based or state-based. For this reason, incorporating both views in a temporal logic in a way that it can be used in specifying real-time system properties is an important research subject in this area. In this paper, we consider both views, and therefore introduce both event and state types in the syntax of ESDL to have more accurate representation of real-time systems.

Another important feature of ESDL is that it incorporates the notion of *duration*, denoting the length of a state (or an event), and *accumulation*, denoting the total duration of a state. These notions are useful for reasoning about time durations of dynamic systems.

We show that the satisfiability problem of ESDL is decidable. We provide a complexity bound for satisfiability, showing that this problem can be solved in NEXPTIME. We also propose a tableau based decision procedure.

Due to its unique syntax ESDL is different from known logics in the literature, which makes this logic interesting to explore (see Section 3.3 for a more detailed discussion). Although ESDL is computationally feasible, it can properly specify important real-time system properties, including punctuality. Actually, we apply ESDL to well-known mine pump [14] and gas burner systems [10], and introduce two verification techniques to check the correctness of ESDL properties. Both systems were studied with some decidable variants of DC [14, 15, 16]; but these variants are based on some abstractions. Since ESDL is defined over genuine continuous intervals, it can capture the system behaviour more accurately.

The plan of this paper is as follows. In Section 2 we will discuss intervals and their relations. In Section 3 the syntax and semantics of ESDL will be presented. In Section 4 we will present the tableau method devised to analyze the complexity of ESDL. In Section 5 we apply the logic ESDL to some safety-critical systems, and analyse the verification problem. In Section 6 we will give some concluding remarks, and discuss some future work.

2. Intervals and Their Relations

In the rest of the paper we represent intervals as pairs of interval bounds which take real numbers. More formally we say that an *interval* is a pair $[t_1, t_2]$ such that $t_1, t_2 \in \mathbb{R}$ and $t_1 \leq t_2$. We denote the set of all intervals $\{[t_1, t_2] : t_1 \leq t_2 \wedge\ t_1, t_2 \in \mathbb{R}\}$ by $\mathcal{I}$, and we use letters $I, J, ...,$ as intervals. It can be simply observed that intervals may be punctual. Intervals can be considered as a closed, bounded, convex and non-empty subset of the real line.

Here, we also define the functions ***beg*** and ***end*** return the beginning and end points of an interval, respectively. For example, $beg([t_1, t_2])$ returns t_1 and $end(J)$ returns t_2.

In [6] Allen introduced thirteen well-known different binary relations between intervals on a linear ordering, which are *before*, *meets*, *overlaps*, *starts*, *during*, *finishes*, *equals*, *finished by*, *includes*, *started by*, *overlapped by*, *met by* and *after*. Given two time intervals J and J', their relative positions can be described by exactly one of the elements of the set of thirteen basic interval relations. Namely, J before J' is denoted by $R^b(J, J')$; J after J' is denoted by $R^{\overline{b}}(J, J')$; J meets J' is denoted by $R^m(J, J')$; J met-by J' is denoted by $R^{\overline{m}}(J, J')$; J overlaps J' is denoted by $R^o(J, J')$; J overlapped-by J' is denoted by $R^{\overline{o}}(J, J')$; J during J' is denoted by $R^d(J, J')$; J includes J' is denoted by $R^{\overline{d}}(J, J')$; J starts J' is denoted by $R^s(J, J')$; J started-by J' is denoted by $R^{\overline{s}}(J, J')$; J finishes J' is denoted by $R^f(J, J')$; J finished-by J' is denoted by $R^{\overline{f}}(J, J')$; and J equals J' is denoted by $R^e(J, J')$.

We can use any combination of the basic relations to get more complex relations. Let $r_1, ..., r_n \in \{b, m, d, s, f, e, o, \overline{b}, \overline{m}, \overline{d}, \overline{s}, \overline{f}, \overline{o}\}$ for $1 \leq n \leq 13$ such that $r_1 \neq ... \neq r_n$. An atomic formula of the form $J\ R^{r_1 \cdots r_n}\ J'$ is satisfied by an interpretation iff $J\ R^{r_i}\ J'$ is satisfied by the same interpretation for some i, $1 \leq i \leq n$. In other words, $J\ R^{r_1 \cdots r_n}\ J' \equiv J\ R^{r_1} \vee ... \vee R^{r_n}\ J'$.

As an example, we define a new relation called *intersection*. We informally say J "intersects" J' iff J either "during", "starts", "finishes", "equal", "overlaps", "includes", "started-by", "finished-by" or "overlapped-by" J'. This can be formally translated into that $J\ R^{dsfeo\overline{d}\overline{s}\overline{f}\overline{o}}\ J'$ iff $J\ (R^d \vee R^s \vee R^f \vee R^e \vee R^o \vee R^{\overline{d}} \vee R^{\overline{s}} \vee R^{\overline{f}} \vee R^{\overline{o}})\ J'$. We will use a more elegant symbol R^i to denote the intersection relation (i.e. $R^i \equiv R^{dsfeo\overline{d}\overline{s}\overline{f}\overline{o}}$).

3. ESDL

As described in Section 1, our logical framework ESDL should express *event* and *state*-based properties and state durations to model the behaviour of safety-critical systems more accurately. The intuition behind the difference between events and states can be summarised as follows [17]:

The sentences which describe state-types have the *homogenity* property. That is, states have the property that whenever they are true over an interval I, they are true over every subinterval of I. Thus, if the weather is hot during the entire day, then it is also hot in the afternoon. On the other hand, events are not *homogenous*. Events have the property that whenever they are true over an interval I, they are false over some subinterval of I, because they are not completed yet. Therefore, if an accident has happened over an interval I, then we cannot say that the accident has happened during a (strict) subinterval of I as the accident is not finished yet.

In order to express this semantic difference in system properties, we separate events and states in the language. Another important design requirement for safety-critical systems is to express state durations, such as "the total duration of a state should not exceed a certain threshold", and critical response properties, such as "a state should be followed by another state within a certain period of

time". Below, we give some example properties that we would like to specify in ESDL:

P1: `The smoke should not last more than 10 seconds.`
P2: `If some smoke is detected, then the alarm should go off in 30 seconds.`
P3: `If the alarm has been ringing for 60 seconds, then it must have lasted for 30 seconds.`
P4: `If the gas tank has exploded in 2 seconds, then the explosion could have occurred in 1 second.`
P5: `While a warning signal is being received, another warning signal (possibly same) has been received during this period.`
P6: `During a period where some smoke is detected, another smoke is detected.`

P1 and P2 are the state duration and critical response properties, respectively. P3 is a typical example of the homogeneity property for states, whereas P4 is an anti-homogeneity property for events. P5 is a good example to show how events (possibly same) can be nested, which cannot be done in most of the interval logics discussed above. P6 shows a typical violation in using states. Namely, this property requires that while a state is being hold, the same state holds again. However, states are already maximal, and two states which are same cannot be nested. Not surprisingly, P1, P2, P3 and P5 are expected to be satisfiable, but P4 and P6 cannot be satisfied.

3.1. Syntax and Semantics

ESDL is interpreted over a linear time flow, where finitely many events and states can occur over bounded-time continuous intervals, which are *primitive* objects of the model. ESDL incorporates the notion of *duration*, denoting the length of a state (or an event), and *accumulation*, denoting the total duration of a state.

In the sequel, let $\mathcal{E}$ and $\mathcal{S}$ be finite sets. We refer to $\mathcal{E}$ as the set of *event atoms* and $\mathcal{S}$ as the set of *state atoms*. $e \in \mathcal{E}$ denotes an event atom and $s \in \mathcal{S}$ denotes a state atom.

Definition 3.1. Let $e \in \mathcal{E}$ be an event atom, $s \in \mathcal{S}$ be a state atom, ϕ, ψ be ESDL formulas, k, d be constants, ℓ be the duration symbol, $r \in \{d, \overline{d}, s, \overline{s}, f, \overline{f}, o, \overline{o}, e\}$ and $\sim \in \{<, >, \leq, \geq, =\}$. Assume $\sigma = e \sim k$ and $\theta = s \sim k$. The logic ESDL is defined by induction as follows:

$$\begin{aligned}\phi ::= \quad & \top \mid \ell \sim k \mid \int s \sim k \mid \textstyle\int_d s \leq k \mid \langle\sigma\rangle\phi \mid [\sigma]\phi \mid \langle\theta\rangle^r\phi \mid [\theta]^r\phi \mid \\ & \langle\theta\rangle^r_<\phi \mid [\theta]^r_<\phi \mid \langle\theta\rangle^r_>\phi \mid [\theta]^r_>\phi \mid \neg\phi \mid \phi \wedge \psi\end{aligned}$$

Here, e (resp. s) denotes the time length of e (resp. s). The connectives $\Rightarrow$ and $\Leftrightarrow$ can be defined in usual way. For simplicity, we will denote the special cases $\langle e \geq 0\rangle$ and $[e \geq 0]$ as $\langle e\rangle$ and $[e]$, respectively. Similarly, we will denote $\langle s \geq 0\rangle$ and $[s \geq 0]$ as $\langle s\rangle$ and $[s]$, respectively. Also, in the sequel, we call

$[\sigma]\phi, [\theta]^r\phi$ |, $[\theta]^r_<\phi$ and $[\theta]^r_>\phi$ as *universal formulas*, and $\langle\sigma\rangle\phi, \langle\theta\rangle^r\phi, \langle\theta\rangle^r_<\phi$ and $\langle\theta\rangle^r_>\phi$ as *existential formulas*.

Definition 3.2. Let $\mathcal{I}$ be the set of all bounded, closed, convex and non-empty intervals of real numbers, $\mathcal{E}$ be a finite set of event atoms, and $\mathcal{S}$ be a finite set of state atoms. An ESDL *model* $\mathcal{M}$ is a *finite* subset of $(\mathcal{I} \times \mathcal{E}) \cup (\mathcal{I} \times \mathcal{S})$ such that for all $J, J' \in \mathcal{I}$ if $\langle J, s\rangle$, $\langle J', s\rangle \in \mathcal{M}$, then $J \cap J' = \emptyset$.

In ESDL, events and states have different semantics. $\langle J, s\rangle \in \mathcal{M}$ intuitively implies that the state s holds throughout any subinterval J' of J. From the definition of an ESDL model $\mathcal{M}$ we can easily conclude that if $\langle J, s\rangle \in \mathcal{M}$, then there is no interval $J' \subseteq J$ or $J \subseteq J'$ such that $\langle J', s\rangle \in \mathcal{M}$ (That is, we do not allow a nested occurrence of the same state.) We can think the interval J as the *maximal* interval at which the state s holds. However, intervals at which any event occurs cannot be considered as maximal intervals. That is, one can define a nested occurrence of event atoms. This is stated more formally as follows: If $\langle J, e\rangle \in \mathcal{M}$, then it is possible that there is an interval $J' \subseteq J$ or $J \subseteq J'$ such that $\langle J', e\rangle \in \mathcal{M}$.

Definition 3.3. Let $J, I \in \mathcal{I}$ be the intervals $[t_1, t_2]$ and $[t'_1, t'_2]$, respectively, with $J \cap I \neq \emptyset$. The partial functions ***inter***, ***init*** and ***fin*** are defined as follows: $inter(J, I)$ denotes the interval $[max(t_1, t'_1), min(t_2, t'_2)]$; $init(J, I)$ denotes the interval $[min(t_1, t'_1), max(t_1, t'_1)]$; and $fin(J, I)$ denotes the interval $[min(t_2, t'_2), max(t_2, t'_2)]$.

Definition 3.4. Let $\mathcal{M}$ be an ESDL model and $I \in \mathcal{I}$. The formal semantics of ESDL formulas is then defined as follows:

$\mathcal{M} \models_I \ell \sim k$ iff $|I| \sim k$

$\mathcal{M} \models_I \int s \sim k$ iff $\sum\{|J \cap I| : \langle J, s\rangle \in \mathcal{M}$ and $R^i(J, I)\} \sim k$

$\mathcal{M} \models_I \int_d s \leq k$ iff $\forall J \in \{J : J = I$ or $\langle J, s\rangle \in \mathcal{M}$ and $R^i(J, I)\}$
$\sum\{|K \cap I| : \langle K, s\rangle \in \mathcal{M}$ and $R^i(K, I')$ s.t. $I' = [beg(J), beg(J) + d]\} \leq k$

$\mathcal{M} \models_I \langle e \sim k\rangle\phi$ iff $\exists J \subseteq I$ such that $\langle J, e\rangle \in \mathcal{M}$, $|J| \sim k$ and $\mathcal{M} \models_J \phi$;

$\mathcal{M} \models_I [e \sim k]\phi$ iff $\forall J \subseteq I$ $\langle J, e\rangle \in \mathcal{M}$ and $|J| \sim k$ imply $\mathcal{M} \models_J \phi$;

$\mathcal{M} \models_I \langle s \sim k\rangle^r\phi$ iff $\exists J$ such that $\langle J, s\rangle \in \mathcal{M}$, $R^r(J, I)$, $|J| \sim k$ and
$\mathcal{M} \models_{inter(J,I)} \phi$;

$\mathcal{M} \models_I [s \sim k]^r\phi$ iff $\forall J$ $\langle J, s\rangle \in \mathcal{M}$, $R^r(J, I)$ and $|J| \sim k$ imply $\mathcal{M} \models_{inter(J,I)} \phi$;

$\mathcal{M} \models_I \langle s \sim k\rangle^r_<\phi$ iff $\exists J$ such that $\langle J, s\rangle \in \mathcal{M}$, $R^r(J, I)$, $|J| \sim k$ and
$\mathcal{M} \models_{fin(J,I)} \phi$;

$\mathcal{M} \models_I [s \sim k]^r_<\phi$ iff $\forall J$ $\langle J, s\rangle \in \mathcal{M}$, $R^r(J, I)$ and $|J| \sim k$ imply $\mathcal{M} \models_{fin(J,I)} \phi$;

$\mathcal{M} \models_I \langle s \sim k\rangle^r_>\phi$ iff $\exists J$ such that $\langle J, s\rangle \in \mathcal{M}$, $R^r(J, I)$, $|J| \sim k$ and
$\mathcal{M} \models_{init(J,I)} \phi$;

$\mathcal{M} \models_I [s \sim k]^r_> \phi$ iff $\forall J\ \langle J, s\rangle \in \mathcal{M}$, $R^r(J, I)$ and $|J| \sim k$ imply $\mathcal{M} \models_{init(J,I)} \phi$;

$\mathcal{M} \models_I \neg\phi$ iff not $\mathcal{M} \models_I \phi$;

$\mathcal{M} \models_I \phi \wedge \psi$ iff $\mathcal{M} \models_I \phi$ and $\mathcal{M} \models_I \psi$.

where $|J|$ denotes the length of the interval J, R^r denotes an interval relation where $r \in \{d, \overline{d}, s, \overline{s}, f, \overline{f}, o, \overline{o}, e\}$ (For instance, R^o denotes the interval relation "overlaps"), and R^i denotes the intersection relation (see Section 2). Here we want the predicate symbol R^r appearing in an ESDL formula to take the intervals J, J', as argument, to have an intersection segment. Therefore, we exclude the interval relations "before", "after", "meets" and "met-by" (i.e. $r \notin \{b, \overline{b}, m, \overline{m}\}$).

To make the formal constructs more comprehensible we provide the informal meanings of ESDL formulas below:

- $\ell \sim k$ holds *iff* the length of the current interval is bounded by $\sim k$.
- $\int s \sim k$ holds *iff* the total duration of s is bounded by $\sim k$.
- $\int_d s \leq k$ holds *iff* the total duration of s is less than or equal to k in any interval of length d (where s holds in the initial segment).
- $\langle e \sim k\rangle\phi$ holds *iff* ϕ holds at some interval (taking place during the current interval) over which the event e occurs and whose length is bounded by $\sim k$.
- $\langle s \sim k\rangle^r\phi$ holds *iff* ϕ holds at some interval (intersecting the current interval) over which the state s holds and whose length is bounded by $\sim k$.
- $\langle s \sim k\rangle^r_<\phi$ holds *iff* the state s holds at some interval J (intersecting the current interval) whose length is bounded by $\sim k$ and ϕ holds at the interval starting immediately after J and ending when the current intervals ends.

3.2. Examples

We now give some examples to show how ESDL can specify quantitative and punctual properties, and how satisfying models look like. Consider the following property:

`Every process p is followed by process q in precisely 5 seconds.`

This property can be stated in ESDL as

$$[\mathtt{p}]^d_< \langle \mathtt{q}\rangle^d_> (\ell = 5).$$

Here, we assume that `p,q` occur "during" the current interval[3]. Figure 1 shows an example model which satisfies this formula. In the figure, I is the current interval. J and L are any intervals which occur "during" I and at which `p` holds.

[3]Recall that for simplicity $\langle s \geq 0\rangle$ and $[s \geq 0]$ are denoted as as $\langle s\rangle$ and $[s]$, respectively.

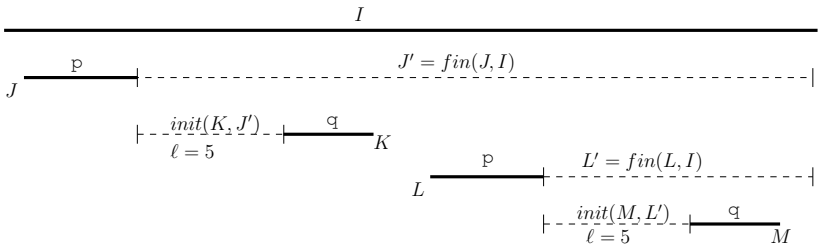

Figure 1: A model satisfying the formula $[\mathtt{p}]^d_< \langle \mathtt{q} \rangle^d_> (\ell = 5)$.

Moreover, $J' = fin(J, I)$ satisfies $\langle \mathtt{q} \rangle^d_> (\ell = 5)$ because there exists an interval K which occurs during J' and at which $\mathtt{q}$ holds, and the distance between J and K is 5 (i.e. $|init(K, J')| = 5$). Similarly, $L' = fin(L, I)$ satisfies $\langle \mathtt{q} \rangle^d_> (\ell = 5)$ because there exists an interval M which occurs during L' and at which $\mathtt{q}$ holds, and the distance between L and M is 5 (i.e. $|init(M, L')| = 5$). So in this model, at the end of every interval with state $\mathtt{p}$ an interval with state $\mathtt{q}$ starts after exactly 5 seconds.

ESDL is very expressive to specify the type of the relation between intervals. For example, in the above example if we wanted to express that $\mathtt{p}$ occurs "during" the interval or "starts" the interval, we would simply write $[\mathtt{p}]^{sd}_< \langle \mathtt{q} \rangle^d_> (\ell = 5)$. Note that we cannot express this property as $[\mathtt{p}]^{sd}_< \langle q \rangle^s_> (\ell = 5)$. Because $\langle \mathtt{q} \rangle^s_>$ means that $\mathtt{q}$ follows $\mathtt{p}$ immediately, which contradicts $\ell = 5$ (asserting that the distance between $\mathtt{q}$ and $\mathtt{p}$ must be 5).

ESDL also allows us to specify durations of states along with the distance in between. For example, the property

`Every process p which lasts at least two seconds is followed by process q, which lasts at most one second, in precisely 5 seconds.`

This property is expressed in ESDL as follows:

$$[\mathtt{p} \geq 2]^d_< \langle \mathtt{q} \leq 1 \rangle^d_> (\ell = 5).$$

Figure 2 illustrates a model which satisfies this formula. In this model, at the end of every interval with state $\mathtt{p}$ and length greater than or equal to 2, an interval with state $\mathtt{q}$ and length less than or equal to 1 starts after exactly 5 seconds. As seen in the figure, an interval with state $\mathtt{q}$ and length less than 2 (e.g. the interval N) is not necessarily followed by an interval with state $\mathtt{q}$.

We can now show how the properties discussed above are formally specified. Below, F1, F2, F3, F4, F5 and F6 are the ESDL formulas expressing the properties P1, P2, P3, P4, P5 and P6, respectively.

F1: $\int Smoke \leq 10$

F2: $[Smoke]^d_< \langle Alarm \rangle^d_> (\ell \leq 30)$

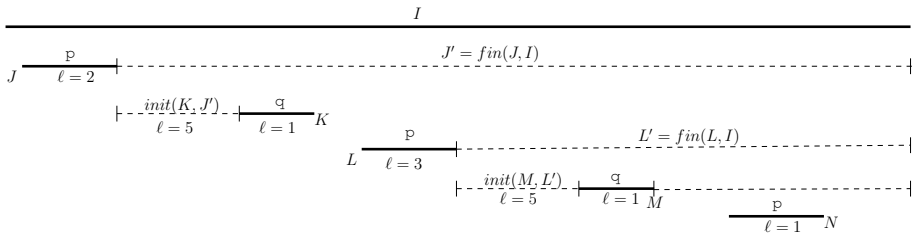

Figure 2: A module satisfying the formula $[\mathtt{p} \geq 2]^d_< \langle \mathtt{q} \leq 1 \rangle^d_> (\ell = 5)$.

F3: $\langle Alarm \geq 60 \rangle^o (\ell = 30)$

F4: $\langle Explode \geq 2 \rangle (\ell = 1)$

F5: $\langle Signal \rangle \langle Signal \rangle \top$

F6: $\langle Smoke \rangle^d \langle Smoke \rangle^d \top$

Above, we consider *Smoke* and *Alarm* as states, because if they are true over an interval, they are also true over every subinterval. Whereas, we consider *Explode* and *Signal* as events as they are not completed in a (strict) subinterval. F1 ensures that the total duration of the state *Smoke* is less than 10 s. F2 requires that the every occurrence of *Smoke* is followed by *Alarm* in less than 30 s. (Note that we assume *Smoke* and *Alarm* hold "during" the current interval). F3 asserts that *Alarm* has started before the current interval and has lasted more than 60 s., and the length of its intersection segment with the current interval is 30. F4 implies that there is an interval whose length is more than 2 at which *Explode* occurs, and the length of this interval is 1 (asserted by $\ell = 1$). Clearly, F4 cannot be satisfied in any model. F5 states that the *Signal* events occur nested within each other. Finally, F6 states that the *Smoke* states occur nested within each other, which violates the condition in Definition 3.2; therefore it is not a satisfiable formula.

3.3. Comparison

In this section, we briefly compare ESDL with well-known related logics in the literature. A formal account is outside the scope of this paper.

The *propositional interval logics* HS [18], CDT [19], PNL [20], PITL [7] and various fragments of these logics do not have metric operators. These logics, therefore, cannot express quantitative (and punctuality) properties. On the other hand, ESDL cannot express certain qualitative properties, such as *unboundedness* that these logics can express.

ESDL is also incomparable with the *first-time intervals logics* ITL [7], DC [10] (and its expressive decidable fragments e.g. [21]), NL [11] and IDL [15]. ITL does not have metric operators. ESDL, however, cannot express some first-order properties that ITL can express. DC, NL and IDL can express all state-based properties that ESDL can express. So, we can say that DC, NL

and IDL subsume ESDL's state-based semantics (In Section 5.5 we define an inductive translation, which maps state-based ESDL formulas into DC formulas). However, these logics do not support some event semantics that ESDL does. For example, the property *"during the event* e *another event* e *occurs"* is not expressible in these logics. ESDL, on the other hand, cannot express properties which require universal quantification at every interval. For example, the statement *"φ is true at every interval"* can be expressed in ITL, DC, NL and IDL; but not in ESDL.

Here, it is worth to mention that one important characteristic of ESDL formulas is the 'quasi-guarded' nature of the quantification they feature. Thus, for example, the formula $\langle e \rangle \phi$ existentially quantifies over intervals satisfying the event e (similarly for universal formulas). So, it does not quantify over all subintervals of the current interval of evaluation without restriction. However, many logics, such as HS, CDT, PNL, DC, NL, ITL and IDL, lack the 'quasi-guarded' character of the quantification that ESDL formulas feature. This feature is very important to achieve decidability. Another important feature for decidability is that ESDL has the finite model property. Although this feature does not always guarantee decidability, together with quasi-guardedness it provides interesting and useful model-theoretic properties for ESDL. On the other hand, this interaction causes a limitation. ESDL cannot specify 'fairness' properties, which prevents us from modeling periodic behaviours of systems.

ESDL is also not comparable to *real-time interval logics* RTL [22], RTIL [8], MTIL [1] and TILCO [9]. These logics cannot express event-based properties, such as *"during the event* e *another event* e *occurs"*, also they do not have modality for 'accumulation'. On the other hand, RTL has a reference to an absolute clock, which ESDL does not. In RTIL, instants can be specified absolutely or relative to the beginning of the current context, which cannot be done in ESDL. In MTIL quantification at every interval is possible; but this cannot be done in ESDL. MTIL rules out equality constraints, which ESDL includes in its syntax. TILCO extends FOL; therefore, TILCO can express some first-order properties that ESDL cannot express.

Above we compared ESDL with various interval logics. In the literature there are also many *point-based* logics. The most influential ones are LTL [23], CTL [24], TCTL [25], RTCTL [26], ML [27], RTTL [28], TPTL [29], XCTL [30] an TRIO [31]. Some of these logics do not have metric operators, e.g. LTL and CTL. Some logics have metric operators but cannot express punctuality, e.g. RTCTL. For those logics that can express punctuality such as TCTL and RTTL, the limitations of point-based logics, discussed in Section 1, apply.

One important point is that almost all of the interval-based logics that we mentioned are undecidable; except PNL, RTIL, MTIL and some fragments of DC, which are decidable. ESDL is one of a few decidable interval logics which can be used in the specification of real-time systems. Also, ESDL has a different syntax than known logics. These characteristics make ESDL interesting to explore.

Finally, we note that ESDL is related to some logics in the literature. In [32] an *event*-based interval logic is introduced for capturing the meanings of

temporal constructions, and in [33] a new fragment of first-order logic with two variables (FO^2), a general framework for *event*-based interval temporal logics, is defined. These logics are convenient for expressing the semantics of natural language constructions; but they cannot be used in formal modeling of real-time and safety-critical requirements. As well as events and event-based constructs, ESDL also introduces states and state-based operators which allow expressing quantitative temporal constraints and *punctuality* timing constraints over continuous intervals. A general framework for ESDL can be found in [34].

4. Decidability and Complexity Analysis

The techniques used in this paper for the decidability analysis are similar to [34, 33]. Since ESDL includes the state-based operators, duration operator and accumulation operator, the formal analysis becomes much more complex. We remark that some definitions and certain parts in this paper are taken verbatim from [34, 33]. We also note that we extend the tableau method (and formal proofs) used in [33]. This tableau method (and formal proofs) can be considered as a special case of the one presented in [34]. Since ESDL incorporates state, duration and accumulation operators, we need to consider Allen's interval algebra. In [33], on the other hand, we have only considered a much simpler case, the subinterval relation.

4.1. ESDL Models

In this section we show that the depth of an ESDL model is polynomially bounded on the length of a given formula φ whose satisfiability is checked.

Definition 4.1. Given an ESDL formula φ and a non-empty model $\mathcal{M}$, the **depth**[4] of $\mathcal{M}$ is the greatest m for which there exist $J_1 \subseteq ... \subseteq J_m$ such that for all i, $1 \leq i \leq m$ and for some $e \in \mathcal{E}$, $\langle J_i, e\rangle \in \mathcal{M}$. The depth of an empty model is defined to be 0.

When we determine the depth of a model, we only consider event types. Since an interval at which a state s holds is a maximal interval, and s holds at all subintervals, it is not involved in the definition of depth. We remark that we construct $\mathcal{M}$ in a such way that if $\langle J, s\rangle \in \mathcal{M}$, then there is no $J' \subseteq J$ or $J \subseteq J'$ such that $\langle J', s\rangle \in \mathcal{M}$.

Lemma 4.2. *Let the number of symbols in a given ESDL formula φ be denoted by $|\varphi|$. For a given model $\mathcal{M}$, and interval I, if $\mathcal{M} \models_I \varphi$, then there exists a model $\mathcal{M}^* \subseteq \mathcal{M}$, with depth at most $O(|\varphi|^2)$, such that $\mathcal{M}^* \models_I \varphi$.*

The proof is given in the Appendix A. The proof is based on reducing the model $\mathcal{M}$ to $\mathcal{M}^*$ by removing redundant elements from $\mathcal{M}$. We then prove that both models entail the same set of subformulas.

[4]The definition is borrowed from [35].

4.2. A Tableau for ESDL

In the following, we define a tableau-based decision procedure for ESDL, and analyze its computational complexity. We build models whose sizes are exponentially bounded. We thus show that the satisfiability problem is decidable, and it is in NEXPTIME.

4.2.1. Preliminary notions

Below we introduce some preliminary notions:

Definition 4.3. A **decorated graph** $\mathcal{G} = (V, E)$ is a directed graph comprising a set V of *nodes* where every node has a *decoration* and a set E of edges which are 2-element subsets of V. For a node $v \in V$, a **decoration** $\lambda(v)$ is a 5-tuple $([b_v, e_v],\ \rho(v), \mathcal{K}(v), \mathcal{L}(v),\ \bar{\mathcal{L}}(v))$, where b_v (e_v) is a *constraint variable* denoting the beginning (ending) of the interval represented by node v, $\rho(v)$ denotes the label of node v (where $\rho(v) \in \mathcal{E} \cup \mathcal{S} \cup \{root, -\}$), and $\mathcal{K}(v)$, $\mathcal{L}(v)$ and $\bar{\mathcal{L}}(v)$ denote a set of subformulas associated with node v.

Definition 4.4. A **successor** of a node v is a node w such that there is an edge from v to w. A **path** is a sequence of nodes $v_1, ..., v_k$ such that for all $1 \leq i < k$, v_{i+1} is a successor of v_i. The **depth** of a node v is the maximum number of edges of a path from the root node to v.

Definition 4.5. A **temporal constraint** is a relation involving constraint variables which denote interval endpoints.

Definition 4.6. A **tableau** for a given formula φ is a tuple $\langle \mathcal{G}, \mathcal{C} \rangle$, where $\mathcal{G}$ denotes a decorated graph, and $\mathcal{C}$ denotes a set of temporal constraints in $\mathcal{G}$.

Definition 4.7. A tableau $\langle \mathcal{G}, \mathcal{C} \rangle$ where $\mathcal{G} = (V, E)$ is **closed** if one of the following conditions hold:

- $\bot \in \mathcal{L}(v)$ for some node $v \in V$,
- $\mathcal{C}$ is not satisfiable,
- The depth of the shortest path $v_0 \rightarrow ... \rightarrow v$ is more than $|\varphi|^2$ for some node $v \in V$ (where v_0 is the root node).

Definition 4.8. A tableau is **open** if it is not closed.

4.2.2. Tableau Method

Given a formula φ, we incrementally and nondeterministically construct a tableau $\langle \mathcal{G}, \mathcal{C} \rangle$, where $\mathcal{G} = (V, E)$ is a decorated graph and $\mathcal{C}$ is a set of linear constraints.

Let φ be a formula to be checked for satisfiability over an interval I_0. The *initial tableau* for φ is the tuple $\langle (v_0, \emptyset), \mathcal{C}_0 \rangle$, where $(v_0, \emptyset)$ is the initial graph with decoration $\lambda(v_0) = ([b_{v_0}, e_{v_0}], \rho(v_0), \mathcal{K}(v_0),\ \mathcal{L}(v_0), \bar{\mathcal{L}}(v_0))$ such that $\rho(v_0) = root$, $\mathcal{K}(v_0) = \{\varphi\}$, $\mathcal{L}(v_0) = \emptyset$, $\bar{\mathcal{L}}(v_0) = \emptyset$, and $\mathcal{C}_0$ is the initial set of temporal constraints such that $\mathcal{C}_0 = \{b_{v_0} = beg(I_0), e_{v_0} = end(I_0), b_{v_0} \leq e_{v_0}\}$. Assume Q denotes the queue of nodes in V awaiting processing. Then, the initial value of Q is $\{v_0\}$.

$\mathcal{G} = (V, E)$ is obtained by expanding the initial node v_0 through successive applications of the *expansion strategy* to existing nodes until no node remains to process, and $\mathcal{C}$ is obtained by expanding the initial constraint set $\mathcal{C}_0$ with temporal constraints in existing nodes. In other words, the expansion strategy is applied to every node in Q until Q becomes empty. When a node is selected, it is removed from Q.

In $\mathcal{G} = (V, E)$, an edge $e \in E$ between two nodes means that the intervals represented by these nodes must intersect. Each node $v \in V$ represents an interval $[b_v, e_v]$, where b_v and e_v denote the beginning and end point of this interval and they take real values. The set of constraints $\mathcal{C}$ contains relations between these variables. For each node at minimum $b_v \leq e_v$ is required, but there are other constraints, such as constraints between variables of different nodes (i.e. constraints between different intervals) and constraints enforcing durational restrictions. For example, the temporal constraint $b_v > b_u, e_v < e_u$ shows an interval relation between the intervals $J_v = [b_v, e_v]$ and $J_u = [b_u, e_u]$, which means that J_v occurs "during" J_u.

We note that the sets $\mathcal{K}(v), \mathcal{L}(v)$ and $\bar{\mathcal{L}}(v)$ contain information about formulas that should be satisfied at the node v. Namely, $\mathcal{K}(v)$ contains a set of subformulas, which is not processed (i.e. nondeterministic selection has not been made), to be satisfied at the node v.[5] $\mathcal{L}(v)$ and $\bar{\mathcal{L}}(v)$ contain a set of subformulas which are in the form of DNF disjuncts to be satisfied at the node v. Namely, they contain nondeterministically selected formulas from the subformulas contained in $\mathcal{K}(v)$. The difference between $\mathcal{L}(v)$ and $\bar{\mathcal{L}}(v)$ is that $\mathcal{L}(v)$ keeps subformulas which are satisfied throughout all executions of the expansion strategy (i.e. serves like a cache); but $\bar{\mathcal{L}}(v)$ is re-instantiated in each execution.

The **expansion strategy** is defined as follows:

$Q := \{v_0\}$
$V := \{v_0\}$
$E := \emptyset$
$\mathcal{C} := \mathcal{C}_0$
while $Q \neq \emptyset$ **and** $\mathcal{G}$ is open **do**
  select $v \in Q$; $Q := Q \setminus \{v\}$
  apply the following rules sequentially;
  abort immediately when $\mathcal{G}$ is no longer open:
    Rule 1: *interval relation rule*
    Rule 2: *DNF rule*
    Rule 3: *universal node expansion rule*
    Rule 4: *existential node expansion rule*
    Rule 5: *duration rule*
    Rule 6: *accumulation rule*
end while

[5] Note that when we say "a formula φ is satisfied at a node v", we mean "a formula φ is satisfied at an interval represented by the node v".

The rules in the expansion strategy are defined as follows. We refer the reader to Appendix B for technical details.

interval relation rule: nondeterministically guesses the interval relation between v and *all* other nodes in the graph. For example, assume v is labeled with a state atom s (i.e. $\rho(v) = s$) and there exists a node $u \in V$ with $\rho(u) = s$ and the subformula $[s \sim k]^d\psi$ is satisfiable at u (i.e. $[s \sim k]^d\psi \in \mathcal{L}(u)$). If the interval relation to be guessed is v "during" u,[6] then we set $\mathcal{C} := \mathcal{C} \cup \{b_v > b_u, e_v < e_u, (e_v - b_v) \sim k\}$ and expand E with a new *edge* from u to v (i.e. $E := E \cup \{u \to v\}$) *if* $\{v \to u\}$ *has not been added previously* (i.e. $v \to u \notin E$). Note that in the interval relation rule, we consider the possibility that $\mathcal{L}(u)$ of an existing node u includes a universal subformula which might force that new subformulas become true at the node v. In our example, we expand $\mathcal{K}(v)$ with ψ, because the formula $[s \sim k]^d\psi \in \mathcal{L}(u)$ forces ψ to be true at v.

DNF rule: converts formulas in $\mathcal{K}(v)$ into their DNF forms and chooses nondeterministically one of the disjuncts. Assume $\mathcal{K}(v) = \{\psi_1, ..., \psi_n\}$ for $n \geq 1$, where *each* ψ_i for $1 \leq i \leq n$ is a subformula (of the given formula). Let the Disjunctive Normal Form (DNF) of ψ_i be $\psi_{i1} \vee ... \vee \psi_{in_i}$ for $n_i \geq 1$. Assume we select the disjunct $\psi_{ij} = \psi_{ij1} \wedge ... \wedge \psi_{ijk}$ for $1 \leq j \leq n_i$ and $k \geq 1$. Set $\bar{\mathcal{L}}(v) = \emptyset$. For *each* ψ_{ijl} for $1 \leq l \leq k$, set $\bar{\mathcal{L}}(v) := \bar{\mathcal{L}}(v) \cup \{\psi_{ijl}\}$ if ψ_{ijl} is not already in $\mathcal{L}(v)$. Set $\mathcal{L}(v) := \mathcal{L}(v) \cup \bar{\mathcal{L}}(v)$ and $\mathcal{K}(v) := \emptyset$.

universal node expansion rule: expands all universal subformulas at the node v (i.e. universal subformulas in $\mathcal{L}(v)$). As a result of applying the universal node expansion rule, some of existing nodes might be forced to satisfy new subformulas. In this case, these nodes will be re-visited, which means the expansion strategy is re-executed for these nodes. Assume the subformula $[s \sim k]^d\psi$ is satisfiable at v (i.e. $[s \sim k]^d\psi \in \mathcal{L}(v)$). Assume also that there exists a node $u \in V$ with label s and the interval relation has already guessed u "during" v. In this case, we expand $\mathcal{K}(u)$ with ψ and set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim k\}$, and $Q := Q \cup \{u\}$. Namely, the universal node expansion rule forces u to satisfy ψ and the expansion strategy is re-executed for this node.

existential node expansion rule: expands all existential subformulas at the node v by creating a new node (or nodes). In the next run, the expansion strategy is applied to the new node(s). Assume $\langle s \sim k\rangle^d\psi \in \mathcal{L}(v)$. The existential node expansion rule adds an immediate successor w with $\rho(w) = s$, $\mathcal{K}(w) = \{\psi\}$, and sets $\mathcal{C} := \mathcal{C} \cup \{b_w > b_v, e_w < e_v, (e_w - b_w) \sim k\}$, $Q := Q \cup \{w\}$, $V := V \cup \{w\}$ and $E := E \cup \{v \to w\}$.

duration rule: checks whether the length of the interval represented by the given node satisfies a certain bound. Assume $\ell \sim k \in \mathcal{L}(v)$ for a given node v. The duration rule sets $\mathcal{C} := \mathcal{C} \cup \{(e_v - b_v) \sim k\}$.

[6] When we refer to an interval relation between two nodes, such as v "during" u, we mean that this interval relation holds between the intervals represented by these nodes.

accumulation rule: checks whether the total duration of a state satisfies a certain condition. Assume $\int s \sim k \in \mathcal{L}(v)$ for a given node v. The accumulation rule guesses the interval relation between v and any node $w \in V$ with $\rho(w) = s$, and updates $\mathcal{C}$ accordingly (e.g. for w "during" v, we set $\mathcal{C} := \mathcal{C} \cup \{b_w > b_v, e_w < e_v\}$). Then, the duration of the intersection segment of v and any w is added up (e.g. $Sum := Sum + min(e_v, e_w) - max(b_v, b_w)$), and the total sum is represented as a constraint $\mathcal{C} := \mathcal{C} \cup \{Sum \sim k\}$.

During the application of the expansion strategy to a node, we need to solve the temporal constraints in $\mathcal{C}$. Remember that each node of the graph represents an interval. For our purposes, we model intervals as pairs of endpoints, which are distinct numbers on the real line. Let $T = \{b_{v_1}, ..., b_{v_n}, e_{v_1}, ..., e_{v_n}\}$ be a set of constraint variables. The constraints of a tableau can be represented as a Simple Temporal Problem [36]. Given that n is the number of variables the complexity of a solution to a STP (if there is any) can be found in $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ space. If the set of temporal constraints in $\mathcal{C}$ is inconsistent, then a solution will not be found, and we say $\mathcal{C}$ is not satisfiable.

In order to avoid infinite paths we need to guarantee the termination of the proposed tableau method. Definition 4.7 gives a suitable *stopping condition* for the tableau procedure. Namely, as soon as one of the conditions holds, we do not need to make any further execution, as we know that the tableau is closed, and the formula is not satisfiable. We therefore apply the expansion strategy only if the tableau is open.

For a given formula φ if there is an open tableau, i.e. none of the stopping conditions is true, then φ is satisfiable, and the satisfying model $\mathcal{M}$ is derived from the tableau. We do this by picking some solution σ, which assigns real values to constraint variables in $\mathcal{C}$. Let $J_v = [\sigma(b_v), \sigma(e_v)]$ be the interval represented by a node $v \in V$. We construct a model $\mathcal{M}$ as follows: $\mathcal{M} = \{\langle J_v, \rho(v)\rangle \mid$ for any $v \in V$ s.t. $\rho(v) \notin \{root, -\}\}$. If the tableau is closed, then φ is unsatisfiable.

Remark 1. In the expansion strategy, Rules 1, 2, 3, 4, 5 and 6 are applied sequentially. Rule 1 must be applied first, because we want to collect all new subformulas (which must be satisfied at a given node v) forced by some other nodes before expanding the graph. Once these subformulas are added to $\mathcal{K}(v)$, we need to calculate the DNF form of $\mathcal{K}(v)$ and select a disjunct upon which the graph will be expanded. After this stage we should expand universal subformulas, because node v might force some visited nodes to satisfy some new subformulas. This should happen before calling existential node expansion rule. We apply duration and accumulation rules last, because we want to check durational constrains once all nodes have been created in each round.

Remark 2. The expansion strategy is not optimal. Namely, it might be the case that same interval constraints can be added into $\mathcal{C}$ several times. For example, the interval relation rule might guess the relation between two nodes even if this relation has already been known. This is due to the non-deterministic nature of the tableau method introduced. The method can be improved to avoid extra

computations by introducing some control variables. But this will only improve the efficiency of the method, it will not reduce the computational complexity we have found.

4.2.3. Application of the Tableau Method

We now apply the proposed tableau method to check the satisfiability of an ESDL formula to make the procedure discussed above clearer. Assume,

$$\begin{array}{lcl} \psi_1 & = & \langle s = 1 \rangle^f \psi \\ \psi_2 & = & \langle s' = 2 \rangle^d \top \\ \psi & = & [s']^o \phi \\ \phi & = & \langle s'' \rangle^d \top \end{array}$$

Let $\varphi = \psi_1 \wedge \psi_2$ be a formula to be checked for satisfiability over an interval I_0. The initial tableau for φ is the tuple $\langle (v_0, \emptyset), \mathcal{C}_0 \rangle$, where v_0 is the initial graph with the decoration $\lambda(v_0) = ([b_{v_0}, e_{v_0}], root, \{\varphi\}, \emptyset, \emptyset)$, and $\mathcal{C}_0 = \{b_{v_0} = beg(I_0), e_{v_0} = end(I_0), b_{v_0} \leq e_{v_0}\}$. Also, the initial value of Q is $\{v_0\}$. We now show how the rules of the expansion strategy is applied to the root node v_0:

$$\downarrow \text{Apply Rule 1 to } v_0$$

v_0 is selected and Q is set to $Q \setminus \{v_0\}$. Rule 1 applies the interval relation rule to v_0. Since v_0 is the only node in the graph, the interval rule does not do anything.

$$\downarrow \text{Apply Rule 2 to } v_0$$

Rule 2 sets $\bar{\mathcal{L}}(v_0) = \mathcal{L}(v_0) = \{\psi_1, \psi_2\}$.

$$\downarrow \text{Apply Rule 3 to } v_0$$

The universal node expansion rule does not do anything.

$$\downarrow \text{Apply Rule 4 to } v_0$$

The existential node expansion rule adds two new nodes v_1 and v_2 (for ψ_1 and ψ_2) with $\lambda(v_1) = ([b_{v_1}, e_{v_1}], s, \{\psi\}, \emptyset, \emptyset)$ and $\lambda(v_2) = ([b_{v_2}, e_{v_2}], s', \{\top\}, \emptyset, \emptyset)$, and sets $Q := Q \cup \{v_1, v_2\}$, $V := V \cup \{v_1, v_2\}$, $E := E \cup \{v_0 \rightarrow v_1, v_0 \rightarrow v_2\}$ and $\mathcal{C} := \mathcal{C}_0 \cup \{b_{v_1} > b_{v_0}, e_{v_1} = e_{v_0}, b_{v_2} > b_{v_0}, e_{v_2} < e_{v_0}, (e_{v_1} - b_{v_1}) = 1, (e_{v_2} - b_{v_2}) = 2\}$ (including the constraints representing v_1 "finishes" v_0, v_2 "during" v_0, $|J_{v_1}| = 1$ and $|J_{v_2}| = 2$).

$$\downarrow \text{Apply Rule 5 \& 6 to } v_0$$

The duration and accumulation rules do not do anything.

$$\downarrow \text{Apply Rule 1 to } v_1$$

v_1 is selected and Q is set to $Q \setminus \{v_1\}$. Rule 1 applies the interval relation rule to v_1. The interval relation rule nondeterministically guesses the interval relation

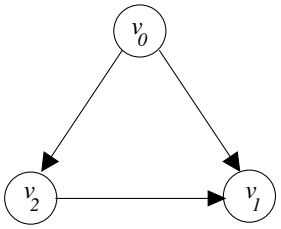

Figure 3: The graph after Rule 1 is applied to node v_1.

between the nodes v_1 and v_0 and v_1 and v_2. Assume the nondeterministic choice between v_1 and v_0 is v_1 "finishes" v_0.[7] In this case, $\mathcal{C}$ is set to $\mathcal{C} \cup \{b_{v_1} > b_{v_0}, e_{v_1} = e_{v_0}\}$. Assume also that the nondeterministic choice between v_1 and v_2 is v_1 "overlapped-by" v_2 (i.e. v_2 "overlaps" v_1). In this case, $\mathcal{C}$ is set to $\mathcal{C} \cup \{b_{v_2} < b_{v_1} < e_{v_2} < e_{v_1}\}$. The interval rule also sets $E := E \cup \{v_2 \to v_1\}$. At this stage, The graph (V, E) becomes as in Figure 3.

$$\downarrow \text{Apply Rule 2 to } v_1$$

Rule 2 sets $\bar{\mathcal{L}}(v_1) = \mathcal{L}(v_1) = \{\psi\}$.

$$\downarrow \text{Apply Rule 3 to } v_1$$

The universal rule is fired since $[s']^o\phi \in \mathcal{L}(v_1)$, $\rho(v_2) = s'$ and v_2 "overlaps" v_1. This rule creates a new node v_3 with $\lambda(v_3) = ([b_{v_3}, e_{v_3}], -, \{\phi\}, \emptyset, \emptyset)$, and sets $Q := Q \cup \{v_3\}$, $V := V \cup \{v_3\}$, $E := E \cup \{v_1 \to v_3, v_2 \to v_3\}$ and $\mathcal{C} := \mathcal{C} \cup \{b_{v_3} = b_{v_1}, e_{v_3} = e_{v_2}\}$.

Rules 4, 5 and 6 do not change anything.

$$\downarrow \text{Apply Rule 1 to } v_2$$

v_2 is selected and Q is set to $Q \setminus \{v_2\}$. Rule 1 applies the interval relation rule to v_2. The interval relation rule nondeterministically guesses the interval relation between v_2 and any other node in V. Assume the interval relation chooses the following relations: v_2 "during" v_0, v_2 "overlaps" v_1 and v_3 "finishes" v_2. In this case, $\mathcal{C}$ is set to $\mathcal{C} \cup \{b_{v_2} > b_{v_0}, e_{v_2} < e_{v_0}\}$ for v_2 "during" v_0 ($\mathcal{C}$ is updated similarly for other interval relations).

$$\downarrow \text{Apply Rule 2 to } v_2$$

Rule 2 sets $\bar{\mathcal{L}}(v_2) = \mathcal{L}(v_2) = \{\top\}$.

Rules 3, 4, 5 and 6 do not change anything.

$$\downarrow \text{Apply Rule 1 to } v_3$$

[7] The interval relation rule could choose another relation which might lead to inconsistency in $\mathcal{C}$; but since our method is nondeterministic, as long as one choice results in a consistent $\mathcal{C}$ we do not need worry about other bad choices.

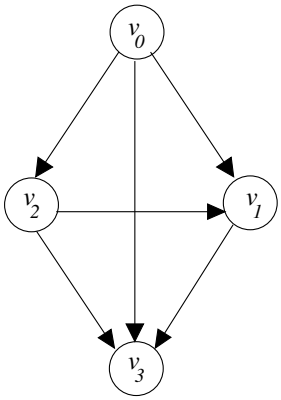

Figure 4: The graph after Rule 1 is applied to node v_3.

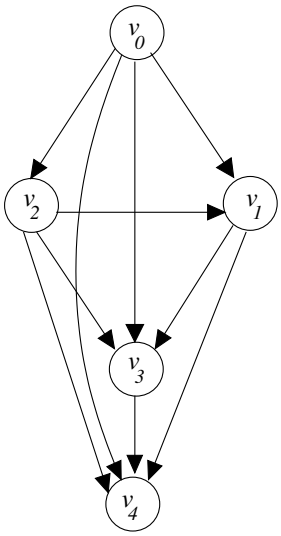

Figure 5: The final graph generated by the tableau method.

v_3 is selected and Q is set to $Q \setminus \{v_3\}$. Rule 1 applies the interval relation rule to v_3. The interval relation rule nondeterministically guesses the interval relation between v_3 and any other node in V. Assume that the interval relations are v_3 "during" v_0, v_3 "starts" v_1 and v_3 "finishes" v_2. In this case, $\mathcal{C}$ is set to $\mathcal{C} \cup \{b_{v_3} > b_{v_0}, e_{v_3} < e_{v_0}, b_{v_3} = b_{v_1}, e_{v_3} < e_{v_1}, b_{v_3} > b_{v_2}, e_{v_3} = e_{v_2}\}$. The interval rule also sets $E := E \cup \{v_0 \to v_3\}$. The graph (V, E) now becomes as in Figure 4.

$$\downarrow \text{Apply Rule 2 to } v_3$$

Rule 2 sets $\bar{\mathcal{L}}(v_3) = \mathcal{L}(v_3) = \{\phi\}$.

$$\downarrow \text{Apply Rule 4 to } v_3$$

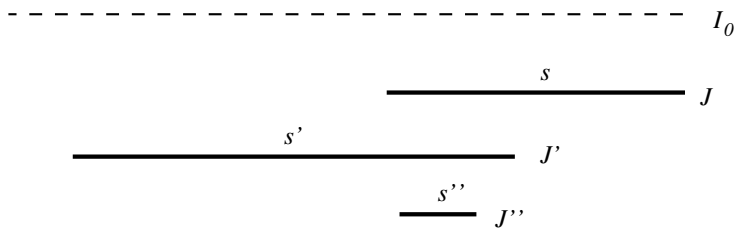

Figure 6: A model for φ. The intervals J,J' and J'' must satisfy the following condition: $|J| = 1$, $|J'| = 2$ and $|J''| \geq 0$. Note that in the figure the given interval I_0 is used to show the relative positions of the intervals J, J' and J''. It is actually not contained in the model.

The existential node expansion rule expands ϕ by adding a new node v_4 with $\lambda(v_4) = ([b_{v_4}, e_{v_4}], s'', \{\top\}, \emptyset, \emptyset)$, and sets $Q := Q \cup \{v_4\}$, $V := V \cup \{v_4\}$, $E := E \cup \{v_3 \rightarrow v_4\}$ and $\mathcal{C} := \mathcal{C} \cup \{b_{v_4} > b_{v_3}, e_{v_4} < e_{v_3}, (e_{v_4} - b_{v_4}) \geq 0\}$.

Rules 3, 5 and 6 do not change anything. The expansion strategy is applied to the node v_4 similarly. The final graph becomes as in Figure 5.

As can be seen, the tableau generated is open. Therefore, a satisfying model $\mathcal{M}$ can be derived from the tableau (suppose we pick some solution for constraint variables in $\mathcal{C}$.) A model for the satisfiable formula φ will be as in Figure 6. Note that the interval represented by a node v with $\rho(v) \in \{root, -\}$ does not appear in the model.

4.3. Soundness-Completeness and Complexity

The proposed tableau method is proved to be sound and complete. The soundness and completeness proofs are given in Appendix C. The upper bound for the complexity of ESDL is in NEXPTIME. We can show this by building a tableau of size $2^{p(|\varphi|)}$ for some fixed polynomial p. The complexity proof is also given in the Appendix C.

5. Application to Safety-Critical Systems

In this section we apply ESDL to some safety-critical systems.

5.1. Abbreviations

Let ϕ be a formula and $\theta = s \sim k$. Assume $\mathcal{R}' = \{s, d, f, e\}$ is a subset $\mathcal{R}$ of thirteen basic interval relations (For the case studies we analyze below, this subset is sufficient to specify the properties. We note that this does not simplify the problem. This subset covers the whole system description.) For simplicity, we will denote $\langle s \geq 0 \rangle$ and $[s \geq 0]$ as $\langle s \rangle$ and $[s]$, respectively. In the following, we will use the following abbreviations:

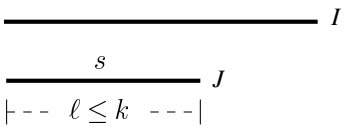

Figure 7: A model satisfying the formula $\lceil s \rceil^k$.

- Let $r_1, ..., r_n \in \mathcal{R}'$ for $1 \leq n \leq 4$ where $r_1 \neq ... \neq r_n$. Then,

$$\begin{array}{lcl} \langle\theta\rangle^{r_1 \ldots r_n} \phi & \equiv & \langle\theta\rangle^{r_1} \phi \vee \cdots \vee \langle\theta\rangle^{r_n} \phi \\ [\theta]^{r_1 \ldots r_n} \phi & \equiv & [\theta]^{r_1} \phi \wedge \cdots \wedge [\theta]^{r_n} \phi \\ \langle\theta\rangle_<^{r_1 \ldots r_n} \phi & \equiv & \langle\theta\rangle_<^{r_1} \phi \vee \cdots \vee \langle\theta\rangle_<^{r_n} \phi \\ [\theta]_<^{r_1 \ldots r_n} \phi & \equiv & [\theta]_<^{r_1} \phi \wedge \cdots \wedge [\theta]_<^{r_n} \phi \end{array}$$

  $\langle\theta\rangle_>^{r_1 \ldots r_n} \phi$ and $[\theta]_>^{r_1 \ldots r_n} \phi$ can be defined similarly.

  For an informal account of the formulas defined above, consider the formulas $\langle\theta\rangle^{sdf} \phi$ and $\langle\theta\rangle_<^{sdf} \phi$. $\langle\theta\rangle^{sdf} \phi$ informally means that θ and ϕ hold in an interval J which either starts, finishes, or takes place during the current interval. $\langle\theta\rangle_<^{sdf} \phi$ informally means that θ holds in an interval J which either starts, finishes, or takes place during the current interval, and ϕ holds at the interval starting immediately after J and ending when the current interval ends.

- $\langle\theta\rangle$ and $[\theta]$ are defined as follows:

$$\begin{array}{lcl} \langle\theta\rangle & \equiv & \langle\theta\rangle^{sdfe} \\ [\theta] & \equiv & [\theta]^{sdfe} \end{array}$$

- $\lceil s \rceil^k$ is true at I iff the existence of a subinterval J of I at which s holds implies that J starts I, and the length of J is less than k. Assume that $|I| > k$. Then, $\lceil s \rceil^k \equiv [s]^{dfe} \bot \wedge [s]^s (\ell \leq k)$.

  $\lceil s \rceil^k$ intuitively implies that a subinterval J of I at which s holds ends within the first k time units of I. A model which satisfies the formula $\lceil s \rceil^k$ is given in Figure 7.

- $\lfloor s \rfloor_k$ is true at I iff there is a subinterval J of I at which s holds, and J begins within the first k time units of I. Assume that $|I| > k$. Then, $\lfloor s \rfloor_k \equiv \langle s \rangle^{se} \top \vee \langle s \rangle_>^{df} (\ell \leq k)$. A model which satisfies the formula $\lfloor s \rfloor_k$ is given in Figure 8.

5.2. DC vs ESDL

Duration Calculus (DC) is a real-time logic for specifying quantitative timing properties of systems. It is originally interpreted over continuous time finitely variable models [10]. Since DC is a highly undecidable logic, there have been

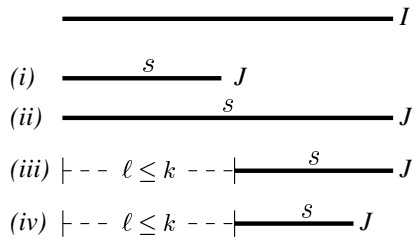

Figure 8: A model satisfying the formula $\lfloor s \rfloor_k$. *(i), (ii), (iii)* and *(iv)* denote different combinations of the intervals J and I. In each case $\lfloor s \rfloor_k$ is true.

several variants of DC which have used certain time abstractions such as, *discrete time* and *sampled time*. A discrete time variant of DC called QDDC was shown to be decidable using finite automata theoretic decision procedure [14]. For automated validity and model checking the tool DCVALID was introduced [14]. QDDC can specify the behaviour of computer programs running on a component with a unique clock. However, in this case it can only specify approximately the behaviour of physical components, and if the system is distributed, it cannot precisely specify the behaviour of the computer programs that run on different components of the system with different clocks [37].

In [15], a sampled time version of DC, called Interval Duration Logic (IDL), was introduced. This logic is also undecidable; but [15] defines partial approaches for validity checking such as abstraction to discrete duration calculus using digitization [38] and bounded validity checking using SMT solvers [39]. Although sampled time logics are closer to automata theoretic models and they may have better decidability properties, using notions such as sampled time can also lead to counter-intuitive behaviour. [16] proposes a reduction method, which reduces the continuous time behaviour of DC to a sampled time version and preserves the validity and the counter examples. However, this provides only a partial method for validity checking of DC, and translated formulas are still checked using discrete time validity checking tool. Various other decidable subsets and approximations of DC have been studied [40, 41, 42]; but they were syntactically very restricted.

Although abstraction methods provide partial solutions in modeling real-time systems, they can only model approximate behaviour of systems. Continuous and dense time domains can provide more accuracy in real-time requirements [43] and can capture exact system behaviour. In this context, finding a decision procedure for the validity problem featuring feasible computational complexity for genuine continuous domains is an open question.

The logic ESDL and the decision procedure we introduced in this paper provide an interesting approach to deal with this problem. Our approach does not use an abstraction method, such as discretization. In ESDL, system states are modeled by intervals which have the finite variability meaning that in any finite interval of time there is only a finite number of discontinuous points. Since we can better reason about the behavioural timing properties of computer pro-

grams, ESDL is preferable to the discrete abstractions of DC in this perspective.

In the following two sections, we will model two well-known mine pump [14] and gas burner systems [10]. Both systems were studied with some decidable variants of DC [14, 15, 16]; but these variants are based on some abstractions. Since ESDL is defined over genuine continuous intervals, it can capture the system behaviour more accurately.

5.3. Gas Burner System

A gas burner is a device to generate a flame to heat up products using gas. The gas burner system is a safety-critical system as excessive leaking of gas may lead to an accident. In this section, we model a gas burner system described in [10], where DC is used.

The critical aspect of the gas burner system is specified by the state *Leak*, which expresses the physical state of gas leak. The real-time requirement and design choices of the gas burner system are described as follows:

Real-time requirement:
For safety, gas must never leak for more than 3 s in any period of 60 s at most.

First design choice:
Any leak should not last longer than 1 s.

Second design choice:
The distance between any two consecutive leaks must be at least 30.

The real-time requirement of the gas burner system can be expressed as an ESDL formula as follows:

$$\mathtt{Req} = \int_{60} Leak \leq 3;$$

The first design choice is expressed in ESDL as follows:

$$\mathtt{Des_1} = [Leak](\ell \leq 1);$$

The following formula is the formulation of the second design choice of the gas burner system in ESDL:

$$\mathtt{Des_2} = [Leak]_{<}[Leak]_{>}([Leak]\bot \Rightarrow \ell \geq 30).$$

The basic control law states that the requirement is implied by the design choices, i.e.

$$\mathtt{Des_1} \wedge \mathtt{Des_2} \Rightarrow \mathtt{Req}.$$

The gas burner system was also modeled in [34], but the verification of the system was not established.

5.4. Mine Pump

"A mine has water seepage which must be removed by operating a pump. If the water is above danger-mark, an alarm must be sounded. There is a high water sensor. In response to water being high, a pump may be operated. The pump must not operate if the water is not high. The mine also has pockets of methane which escape. Presence of methane is detected by a sensor. When there is methane, an alarm must be sounded. Moreover, all electrical activity including the pump must be shut down to prevent explosion" [14].

In this section, by using ESDL, we formalize the mine pump system described in [14], where a logic called *QDDS* is used. A mine pump has the following states:

- $HWater$: Water level is high.
- $DWater$: Water level is dangerous.
- $HMetan$: Methane level is high.
- $Alarm$: Alarm is on.
- $PumpEn$: Pump is enabled.
- $PumpOn$: Pump is operating.

Water Seepage Assumptions:
The high water level occurs before the dangerous water level.

$$\texttt{Asm}_1 \equiv [DWater]_> \langle HWater \rangle \top$$

It takes at least ε seconds for the water level to turn dangerous after reaching the high water level:

$$\texttt{Asm}_2 \equiv [HWater]_< [DWater]_> ([DWater]\bot \Rightarrow \ell \geq \varepsilon)$$

This property specifies a minimum separation of ε time between high and dangerous water levels.

Alarm Control:
The alarm will sound within δ seconds of the water level becoming dangerous. Alarm will persist while the water level is dangerous:

$$\texttt{Alarmcontrol}_1 \equiv [DWater] \langle Alarm \rangle^f_> (\ell \leq \delta)$$

Similarly for Methane:

$$\texttt{Alarmcontrol}_2 \equiv [HMetan] \langle Alarm \rangle^f_> (\ell \leq \delta)$$

Moreover, the alarm will stop within δ seconds, once the methane and the water levels are safe. Let

$$\phi \equiv [HMetan]_< ([HMetan]_> (\ell > \delta) \wedge [DWater]_> (\ell > \delta) \Rightarrow \lceil Alarm \rceil^\delta)$$

$$\psi \equiv [DWater]_{<}([DWater]_{>}(\ell > \delta) \wedge [HMetan]_{>}(\ell > \delta) \Rightarrow \lceil Alarm \rceil^{\delta})$$

Now, we can specify the requirement "the alarm will stop within δ seconds, once the methane and the water levels are safe"as follows:

$$\texttt{Alarmcontrol}_3 \equiv \phi \ \wedge \ \psi$$

Then, $\texttt{Alarmcontrol} \equiv \texttt{Alarmcontrol}_1 \wedge \texttt{Alarmcontrol}_2 \wedge \texttt{Alarmcontrol}_3$.

Pump control:
The pump is started within δ seconds of its being enabled:

$$\texttt{Pumpcontrol}_1 \equiv [PumpEn] \lfloor PumpOn \rfloor_{\delta}$$

The pump is stopped within δ seconds of being disabled:

$$\texttt{Pumpcontrol}_2 \equiv [PumpEn]_{<}([PumpEn]_{>}(\ell > \delta) \Rightarrow \lceil PumpOn \rceil^{\delta})$$

Pump enabling condition: It is safe to operate the pump only when water is high and there is no methane:

$$\texttt{Pumpcontrol}_3 \equiv [HWater]([HMetan]\bot \Rightarrow \langle PumpEn \rangle \top)$$

Then, $\texttt{Pumpcontrol} \equiv \texttt{Pumpcontrol}_1 \wedge \texttt{Pumpcontrol}_2 \wedge \texttt{Pumpcontrol}_3$

Pump Capacity Assumption:
The pump can bring the water level down below the high water level within ϵ seconds:

$$\texttt{Asm}_3 \equiv [PumpOn] \lceil HWater \rceil^{\epsilon}$$

Methane Release Assumptions:
Between two occurrences of methane release there is at least γ seconds:

$$\texttt{Asm}_4 \equiv [HMetan]_{<} [HMetan]_{>} (\ell \geq \gamma)$$

The high methane level lasts at most κ time units:

$$\texttt{Asm}_5 \equiv [HMetan] (\ell \leq \kappa)$$

System Requirement:
The water level never becomes dangerous: $\int DWater = 0$

Verification Condition:
Below a verification condition is established that under the above assumptions the water level never becomes dangerous.

$$\bigwedge_{i=1}^{5} \texttt{Asm}_i \wedge \texttt{Pumpcontrol} \wedge \texttt{Alarmcontrol} \Rightarrow \int DWater = 0.$$

5.5. Verification

In this paper, our main focus is to show that ESDL preserves decidability even if it can express punctuality properties over continuous intervals, and there is a decision procedure, which has a reasonable complexity. In this section, we will introduce two verification techniques, which can be used to check the correctness of ESDL properties.

5.5.1. Deductive Verification

In this section, we consider the *deductive verification* approach to check the correctness of properties. The idea is as follows: Given a specification of a system in ESDL, S, and a property, P, we want to prove that $S \Rightarrow P$ is a *valid* formula. Since ESDL is closed under Boolean operators, we negate the formula we want to show valid, obtaining $\neg(S \Rightarrow P)$ or equivalently $S \wedge \neg P$, and we try to show that the negated formula is *unsatisfiable*. That is, if we add the negation of the property to the specification, then once we establish inconsistency, we know that the non-negated property is a logical consequence of the original specification.

Here, we show how this approach is applied to the verification problem of the gas burner system. Due to space limitation we do not present the tableau construction (we refer the reader to Section 4.2.3 for an example tableau construction); we rather give an intuitive account of the method.

We, namely, want to prove that $\texttt{Des}_1 \wedge \texttt{Des}_2 \wedge \neg\texttt{Req}$ is unsatisfiable. The *universal node expansion rule* adds constraints for the end points of every interval (at which *Leak* holds) such that its length is less than or equal to 1. Similarly, the *universal node expansion rule* and *interval relation rule* extend the constraint set by adding at least 30 between the end point of any interval (at which *Leak* holds) and the beginning of its consecutive interval (at which *Leak* holds). Finally, the *accumulation rule* extends the constraint set to include end point constraints which denote that within every interval (at which *Leak* holds and whose length is 60) the sum of the lengths of intervals (at which *Leak* holds) is greater than 3. This obviously creates a contradiction in the constraint set, because the first two cases together imply that the maximum sum of *Leak* within 60 s intervals is 2. Since the constraints are not satisfied, $\texttt{Des}_1 \wedge \texttt{Des}_2 \wedge \neg\texttt{Req}$ cannot be satisfied.

This approach is suitable for the small systems; but it is not practical if large systems are considered. We therefore propose the following approach.

5.5.2. Reduction to DC Verification

In this section, we consider reducing the verification problem of ESDL into the verification problem of DC. As discussed before, the state-based semantics of ESDL is subsumed by DC. Assume states hold over nonpoint intervals. We can translate some ESDL formulas (ϕ) into DC formulas ($[\phi]_{DC}$) according to the following inductive translation $\phi \mapsto [\phi]_{DC}$:

	ϕ		$[\phi]_{DC}$
R.1	$\ell \sim k$	$\equiv$	$\ell \sim k$
R.2	$\int s \sim k$	$\equiv$	$\int s \sim k$
R.3	$\int_d s \leq k$	$\equiv$	$\Box(\ell \leq d \Rightarrow \int s \leq k)$
R.4	$\langle s \rangle \phi$	$\equiv$	$\Diamond(\lceil\lceil s \rceil\rceil \wedge \phi)$
R.5	$[s]\phi$	$\equiv$	$\Box(\lceil\lceil s \rceil\rceil \Rightarrow \phi)$
R.6	$\langle s \rangle_< \phi$	$\equiv$	$\Diamond(\lceil\lceil s \rceil\rceil ; \phi)$
R.7	$[s]_< \phi$	$\equiv$	$\Box(\lceil\lceil s \rceil\rceil ; true \Rightarrow \lceil\lceil s \rceil\rceil ; \phi)$
R.8	$\langle s \rangle_> \phi$	$\equiv$	$\Diamond(\phi ; \lceil\lceil s \rceil\rceil)$
R.9	$[s]_> \phi$	$\equiv$	$\Box(true ; \lceil\lceil s \rceil\rceil \Rightarrow \phi ; \lceil\lceil s \rceil\rceil)$

We can easily prove that $[\langle s \rangle \phi]_{DC} = [\neg([s]\neg\phi)]_{DC}$, $[\langle s \rangle_< \phi]_{DC} = [\neg([s]_<\neg\phi)]_{DC}$ and $[\langle s \rangle_> \phi]_{DC} = [\neg([s]_>\neg\phi)]_{DC}$.

Above, ";" is the *chop* operator, which is defined as "$\phi ; \psi$ is satisfied over an interval $[b, e]$ iff there exists $b \leq m \leq e$ such that $[b, m]$ satisfies ϕ and $[m, e]$ satisfies ψ" [10].

The DC formula $\lceil\lceil s \rceil\rceil$ intuitively means that s holds almost everywhere in a nonpoint interval. Formally, $\lceil\lceil s \rceil\rceil \equiv \int s = \ell \wedge \ell > 0$. Note that the ESDL formula $[s]\bot$ is equivalent to the DC formula $\lceil\lceil \neg s \rceil\rceil$, because $[s]\bot$ is translated as $\Box(\lceil\lceil s \rceil\rceil \Rightarrow \bot)$, which semantically means that $\lceil\lceil s \rceil\rceil$ does not hold in any interval. This is denoted by the DC formula $\lceil\lceil \neg s \rceil\rceil$.

The verification condition of the gas burner system in ESDL

$$\texttt{Des}_1 \wedge \texttt{Des}_2 \Rightarrow \texttt{Req}$$

then reduces to verification of the following condition in DC:

$$[\texttt{Des}_1]_{DC} \wedge [\texttt{Des}_2]_{DC} \Rightarrow [\texttt{Req}]_{DC}$$

where

$$\begin{array}{lcl} [\texttt{Des}_1]_{DC} & \equiv & \Box(\lceil\lceil Leak \rceil\rceil \Rightarrow \ell \leq 1) \\ [\texttt{Des}_2]_{DC} & \equiv & \Box(\lceil\lceil Leak \rceil\rceil ; \lceil\lceil \neg Leak \rceil\rceil ; \lceil\lceil Leak \rceil\rceil \Rightarrow \ell \geq 30) \\ [\texttt{Req}]_{DC} & \equiv & \Box(\ell \leq 60 \Rightarrow \int Leak \leq 3). \end{array}$$

Now, we prove that by applying the above inductive translation to ESDL formulas $\texttt{Des}_1$, $\texttt{Des}_2$ and $\texttt{Req}$ we obtain the DC formulas $[\texttt{Des}_1]_{DC}$, $[\texttt{Des}_2]_{DC}$ and $[\texttt{Req}]_{DC}$, respectively. The resulting verification problem $[\texttt{Des}_1]_{DC} \wedge [\texttt{Des}_2]_{DC} \Rightarrow [\texttt{Req}]_{DC}$ is exactly same as the verification problem analysed in [10], where the correctness of design choices are proved using the proof system of DC.

Let DC axioms **A1** and **A2** be defined as follows [10]:

$$\textbf{A1} \quad \begin{array}{lcl} (\phi ; \psi) \wedge \neg(\phi ; \varphi) & \Rightarrow & (\phi ; (\psi \wedge \neg\varphi)) \\ (\phi ; \psi) \wedge \neg(\varphi ; \psi) & \Rightarrow & (\phi \wedge \neg\varphi) ; \psi \end{array}$$

$$\textbf{A2} \quad (\phi ; \psi) ; \varphi \Leftrightarrow \phi ; (\psi ; \varphi)$$

Below we prove that $[\langle s\rangle\phi]_{DC} = [\neg([s]\neg\phi)]_{DC}$ and $[\langle s\rangle_<\phi]_{DC} = [\neg([s]_<\neg\phi)]_{DC}$:

$$
\begin{array}{lcll}
[\langle s\rangle\phi]_{DC} & \equiv & [\neg([s]\neg\phi)]_{DC} & \\
 & \equiv & \neg\Box(\lceil\!\lceil s\rceil\!\rceil \Rightarrow \phi) & \textbf{R5} \\
 & \equiv & \Diamond(\lceil\!\lceil s\rceil\!\rceil \wedge \phi) & \\
 & & & \\
[\langle s\rangle_<\phi]_{DC} & \equiv & [\neg([s]_<\neg\phi)]_{DC} & \\
 & \equiv & \neg\Box(\lceil\!\lceil s\rceil\!\rceil; true \Rightarrow \lceil\!\lceil s\rceil\!\rceil; \neg\phi) & \textbf{R7} \\
 & \equiv & \Diamond(\lceil\!\lceil s\rceil\!\rceil; true \wedge \neg(\lceil\!\lceil s\rceil\!\rceil; \neg\phi)) & \\
 & \equiv & \Diamond(\lceil\!\lceil s\rceil\!\rceil; \phi) & \textbf{A1}
\end{array}
$$

We now prove that by applying the inductive translation to ESDL formulas $\texttt{Des}_1$, $\texttt{Des}_2$ and $\texttt{Req}$ we obtain the DC formulas $[\texttt{Des}_1]_{DC}$, $[\texttt{Des}_2]_{DC}$ and $[\texttt{Req}]_{DC}$, respectively:

$$
\begin{array}{lcll}
\texttt{Req} & \equiv & \int_{60} Leak \leq 3 & \\
[\texttt{Req}]_{DC} & \equiv & \Box(\ell \leq 60 \Rightarrow \int Leak \leq 3) & \textbf{R3}
\end{array}
$$

$$
\begin{array}{lcll}
\texttt{Des}_1 & \equiv & [Leak]\,(\ell \leq 1) & \\
[\texttt{Des}_1]_{DC} & \equiv & \Box(\lceil\!\lceil Leak\rceil\!\rceil \Rightarrow \ell \leq 1) & \textbf{R5}
\end{array}
$$

Let L denote $Leak$, $\phi = [L]\bot \Rightarrow \ell \geq 30$ and $[\phi]_{DC} = \lceil\!\lceil\neg L\rceil\!\rceil \Rightarrow \ell \geq 30$

$$
\begin{array}{lcll}
\texttt{Des}_2 & \equiv & [L]_<([L]_>\phi) & \\
[\texttt{Des}_2]_{DC} & \equiv & \Box\left(\lceil\!\lceil L\rceil\!\rceil; true \Rightarrow \lceil\!\lceil L\rceil\!\rceil; (true; \lceil\!\lceil L\rceil\!\rceil \Rightarrow [\phi]_{DC}; \lceil\!\lceil L\rceil\!\rceil)\right) & \textbf{R5,R7,R9} \\
 & \equiv & \Box\left(\lceil\!\lceil L\rceil\!\rceil; true \Rightarrow \lceil\!\lceil L\rceil\!\rceil; \neg\left(true; \lceil\!\lceil L\rceil\!\rceil \wedge \neg\left([\phi]_{DC}; \lceil\!\lceil L\rceil\!\rceil\right)\right)\right) & \\
 & \equiv & \Box\left(\lceil\!\lceil L\rceil\!\rceil; true \Rightarrow \lceil\!\lceil L\rceil\!\rceil; \neg\left(\neg[\phi]_{DC}; \lceil\!\lceil L\rceil\!\rceil\right)\right) & \textbf{A1} \\
 & \equiv & \Box\left(\neg\left(\lceil\!\lceil L\rceil\!\rceil; true \ \wedge\ \neg\left(\lceil\!\lceil L\rceil\!\rceil; \neg\left(\neg[\phi]_{DC}; \lceil\!\lceil L\rceil\!\rceil\right)\right)\right)\right) & \\
 & \equiv & \Box\left(\neg\left(\lceil\!\lceil L\rceil\!\rceil; \left(\neg[\phi]_{DC}; \lceil\!\lceil L\rceil\!\rceil\right)\right)\right) & \textbf{A1} \\
 & \equiv & \Box\left(\neg\left(\lceil\!\lceil L\rceil\!\rceil; \left(\lceil\!\lceil\neg L\rceil\!\rceil \wedge \ell < 30; \lceil\!\lceil L\rceil\!\rceil\right)\right)\right) & \\
 & \equiv & \Box\left(\neg\left(\lceil\!\lceil L\rceil\!\rceil; \left(\lceil\!\lceil\neg L\rceil\!\rceil; \lceil\!\lceil L\rceil\!\rceil\right) \wedge \ell < 30\right)\right) & \text{Proved in [10]} \\
 & \equiv & \Box\left(\neg\left(\lceil\!\lceil L\rceil\!\rceil; \lceil\!\lceil\neg L\rceil\!\rceil; \lceil\!\lceil L\rceil\!\rceil \wedge \ell < 30\right)\right) & \textbf{A2} \\
 & \equiv & \Box\left(\lceil\!\lceil L\rceil\!\rceil; \lceil\!\lceil\neg L\rceil\!\rceil; \lceil\!\lceil L\rceil\!\rceil \Rightarrow \ell \geq 30\right) &
\end{array}
$$

6. Conclusion and Future Work

In this paper, we have proposed the logic ESDL, which is defined over continuous-time intervals and can express punctuality timing constraints. The proposed logic is decidable, and has a reasonable complexity. We have presented a decision procedure for checking validity. We have showed that although ESDL is computationally feasible, it can properly specify some properties which were originally specified by very expensive logics or their reductions based on some abstractions. We have also proposed two techniques to verify ESDL properties.

ESDL uses intervals as primitive objects of time model. Unlike many other interval logics, we do not translate ESDL into a point-based variant, and we therefore try to minimise semantic restrictions. But we can still acquire decidability and a reasonable complexity.

The future research will include defining an axiomatic system, analyzing model checking problem and developing a tool for the automated verification of ESDL properties.

Acknowledgement

The author would like to thank Dr. Ian Pratt-Hartmann for his supervision of the work in [34], guidance and invaluable comments.

References

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

[2] R. Alur, T. A. Henzinger, Logics and models of real time: A survey, in: Proceedings of the Real-Time: Theory in Practice, REX Workshop, Springer-Verlag, 1992, pp. 74–106.

[3] P. Bouyer, N. Markey, J. Ouaknine, J. Worrell, The cost of punctuality, in: Proceedings of the 22nd Annual IEEE Symposium on Logic in Computer Science, LICS '07, IEEE Computer Society, 2007, pp. 109–120.

[4] J. Ouaknine, J. Worrell, On the decidability of metric temporal logic, in: Proceedings 20th Annual IEEE Symposium on Logic in Computer Science, IEEE Computer Society Press, 2005, pp. 188–197.

[5] J. Ouaknine, J. Worrell, On the decidability and complexity of metric temporal logic over finite words, Logical Methods in Computer Science 3 (2007) 27 pages.

[6] J. F. Allen, Maintaining knowledge about temporal intervals, Communications of the ACM 26 (1983) 832–843.

[7] B. Moszkowski, Reasoning about Digital Circuits, Ph.D. thesis, Computer Science Department, Stanford University, 1983.

[8] R. Razouk, M. Gorlick, Real-time interval logic for reasoning about executions of real-time programs, SIGSOFT Softw. Eng. Notes 14 (1989) 10–19.

[9] R. Mattolini, P. Nesi, Using tilco for specifying real-time systems, in: Proceedings of the Second IEEE International Conference on Engineering of Complex Computer Systems, Chapman and Hall, 1996, pp. 18–25.

[10] C. Zhou, M. R. Hansen, Duration Calculus: A Formal Approach to Real-Time Systems (Monographs in Theoretical Computer Science, An EATCS Series), Springer Verlag, 2004.

[11] Z. Chaochen, C. Hoare, A. P. Ravn, A calculus of durations, Information Processing Letters 40 (1991) 269–276.

[12] Z. Chaochen, M. R. Hansen, An adequate first order interval logic, in: Revised Lectures from the International Symposium on Compositionality: The Significant Difference, COMPOS'97, Springer-Verlag, 1998, pp. 584–608.

[13] E. Kindler, T. Vesper, ESTL: A temporal logic for events and states, in: ICATPN'98, LNCS, 1998, pp. 365–384.

[14] P. K. Pandya, Specifying and deciding quantified discrete-time duration calculus formulas using DCVALID, in: Proc. Real-Time Tools, RT-TOOLS'2001, p. 16 pages.

[15] P. K. Pandya, Interval duration logic: Expressiveness and decidability, Electr. Notes Theor. Comput. Sci. 65 (2002) 254–272.

[16] P. Pandya, S. Krishna, K. Loya, On sampling abstraction of continuous time logic with durations, in: O. Grumberg, M. Huth (Eds.), Tools and Algorithms for the Construction and Analysis of Systems, volume 4424 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2007, pp. 246–260.

[17] J. F. Allen, G. Ferguson, Actions and events in interval temporal logic, Journal of Logic and Computation 4 (1994) 531–579.

[18] J. Y. Halpern, Y. Shoham, A propositional modal logic of time intervals, Journal of the ACM 38 (1991) 935–962.

[19] Y. Venema, A modal logic for choppping intervals, Journal of Logic and Computation 1 (1991) 453–476.

[20] V. Goranko, A. Montanari, G. Sciavicco, Propositional interval neighbourhood temporal logics, Journal of Universal Computer Science 9 (2003) 1137–1167.

[21] M. Fränzle, M. R. Hansen, Deciding an interval logic with accumulated durations, in: TACAS'07: Proceedings of the 13th international conference on Tools and algorithms for the construction and analysis of systems, Springer-Verlag, 2007, pp. 201–215.

[22] F. Jahanian, A. K. Mok, Safety analysis of timing properties in real-time systems, IEEE Transactions on Software Engineering SE-12 9 (1986) 890–904.

[23] A. Pnueli, The temporal logic of programs, in: Proceedings of the 18th Annual IEEE Symposium on Foundations of Computer Science, IEEE Computer Society Press, 1977, pp. 46–57.

[24] E. M. Clarke, E. A. Emerson, Design and synthesis of synchronization skeletons using branching-time temporal logic, in: Logic of Programs, Workshop, Springer-Verlag, 1982, pp. 52–71.

[25] R. Alur, C. Courcoubetis, D. L. Dill, Model checking for real-time systems, in: Proceedings 5th Conference on Logic in Computer Science, IEEE Computer Society Press, 1990, pp. 12–21.

[26] E. A. Emerson, A. K. Mok, A. P. Sistla, J. Srinivasan, Quantitative temporal reasoning, in: Proceedings of the 2nd International Workshop on Computer Aided Verification, CAV '90, Springer-Verlag, 1991, pp. 136–145.

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

[28] J. S. Ostroff, W. Wonham, Modeling and verifying real-time embedded computer systems, in: Proceedings of the 8th IEEE Real-Time Systems Symposium, IEEE Computer Society Press, 1987, pp. 124–132.

[29] R. Alur, T. A. Henzinger, Real-time logics: Complexity and expressiveness, in: Proceedings of the 5th Annual IEEE Symposium on Logic in Computer Science, IEEE Computer Society Press, 1990, pp. 390–401.

[30] E. Harel, O. Lichtenstein, A. Pnueli, Explicit clock temporal logic, in: Proceedings of the 5th Annual Symposium on Logic in Computer Science, IEEE Computer Society Press, 1990, pp. 402–413.

[31] C. Ghezzi, D. Mandrioli, A. Morzenti, Trio: A logic language for executable specifications of real-time systems, Journal of Systems and Software 12 (1990) 107–123.

[32] S. Konur, An interval logic for natural language semantics, in: Advances in Modal Logic, College Publications, 2008, pp. 177–191.

[33] S. Konur, An event-based fragment of first-order logic over intervals, Journal of Logic, Language and Information 20 (2011) 49–68.

[34] S. Konur, An Interval Temporal Logic for Real-time System Specification, Ph.D. thesis, Department of Computer Science, University of Manchester, UK, 2008.

[35] I. Pratt-Hartman, Temporal prepositions and their logic, Artificial Intelligence 166 (2005) 1–36.

[36] R. Dechter, I. Meiri, J. Pearl, Temporal constraint networks, Artificial Intelligence 49 (1991) 61–95.

[37] D. V. Hung, K. K. Il, Verification via digitized models of real-time hybrid systems, in: Proceedings of the Third Asia-Pacific Software Engineering Conference, APSEC '96, IEEE Computer Society, 1996, pp. 4–15.

[38] G. Chakravorty, P. Pandya, Digitizing interval duration logic, in: J. Hunt, WarrenA., F. Somenzi (Eds.), Computer Aided Verification, volume 2725 of *Lecture Notes in Computer Science*, Springer, 2003, pp. 167–179.

[39] B. Sharma, P. K. Pandya, S. Chakraborty, Bounded validity checking of interval duration logic, in: Proceedings of the 11th international conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS'05, Springer-Verlag, 2005, pp. 301–316.

[40] Z. Chaochen, Z. Jingzhong, Y. Lu, L. Xiaoshan, Linear duration invariants, in: H. Langmaack, W.-P. Roever, J. Vytopil (Eds.), Formal Techniques in Real-Time and Fault-Tolerant Systems, volume 863 of *Lecture Notes in Computer Science*, Springer, 1994, pp. 86–109.

[41] A. Bouajjani, Y. Lakhnech, R. Robbana, From duration calculus to linear hybrid automata, in: Proceedings of the 7th International Conference on Computer Aided Verification, Springer-Verlag, 1995, pp. 196–210.

[42] M. Fränzle, M. R. Hansen, Efficient model checking for duration calculus?, Int. J. Software and Informatics 3 (2009) 171–196.

[43] Y. Hirshfeld, A. M. Rabinovich, Logics for real time: Decidability and complexity, Fundam. Inform. 62 (2004) 1–28.

APPENDIX A

Proof. (Lemma 4.2) Assume φ is in the desired *normal* form. An ESDL formula is normalized to the desired form by moving $\neg$ inwards. Every ESDL formula is logically equivalent to one in which $\neg$ appears only in subformula of the form $\bot$ $(= \neg\top)$. In an ESDL formula $\neg$ can be moved inwards as follows:

$$\begin{array}{lcl}
\neg\top & \equiv & \bot \\
\neg\ell \sim k & \equiv & \ell \sim' k \\
\neg \int s \sim k & \equiv & \int s \sim' k \\
\neg \int_d s \leq k & \equiv & \int_d s > k \\
\neg\langle e \sim k\rangle\psi & \equiv & [e \sim k]\neg\psi \\
\neg[e \sim k]\psi & \equiv & \langle e \sim k\rangle\neg\psi \\
\neg\langle s \sim k\rangle^r_<\psi & \equiv & [s \sim k]^r_<\neg\psi \\
\neg[s \sim k]^r_<\psi & \equiv & \langle s \sim k\rangle^r_<\neg\psi
\end{array}$$

where $\sim \in \{<, >, \leq, \geq, =\}$ and $\sim'$ is defined as the corresponding *inverted* operator of $\sim$. Namely, $\sim'$ negates the operator $\sim$, e.g. the inverted operator of $<$ is $\geq$. The cases where $\langle s \sim k\rangle^r\psi$ and $\langle s \sim k\rangle^r_>\psi$, $[s \sim k]^r\psi$ and $[s \sim k]^r_>\psi$ are dealt with similarly.

Let φ be an ESDL formula which has the form above. We define $L_e(J)$[8] as follows:

$$L(J) = \{\psi \mid \psi \text{ is a subformula of } \varphi \text{ s.t. } \mathcal{M} \models_J \psi\}$$

$$L_e(J) = L(J) \setminus \bigcup \{L(K) \mid K \subset J,\ \langle K, e\rangle \in \mathcal{M}\}$$

Intuitively, $L(J)$ records the subformulas of φ which are true at an interval J. If we look at the definition, we can see that $L_e(J)$ records the subformulas of φ which are true at an interval J, except the subformulas which are true at some subinterval K of J with $\langle K, e\rangle \in \mathcal{M}$. We say that a pair $\langle J, e\rangle \in \mathcal{M}$ is *redundant* if $L_e(J) = \emptyset$.

We now reduce the model $\mathcal{M}$ to $\mathcal{M}^*$ by removing redundant pairs:

$$\mathcal{M}^* = \mathcal{M} \setminus \{\langle J, e\rangle \mid \langle J, e\rangle \ \textit{is redundant}\}$$

Let m be the number of event atoms occurring in φ, m' be the number of state atoms occurring in φ, and n be the number of subformulas of φ. If $J \subseteq J'$ such that $\langle J, e\rangle \in \mathcal{M}$ and $\langle J', e\rangle \in \mathcal{M}$, then $L_e(J)$ and $L_e(J')$ are disjoint. That is, the length of a chain of the intervals at which e occurs is bounded by the number of the subformulas of φ in which e is mentioned. The length of a chain of the intervals at which a state s holds is 1 since we only consider maximal intervals at which states hold. Therefore, $\mathcal{M}^*$ is bounded by $m.n + m'$.

[8] The definition is borrowed from [35].

Since $m < |\varphi|$, we can conclude that $m' < |\varphi| - m$. We also know that $n < |\varphi|$. It easily follows that the depth of $\mathcal{M}^*$ is bounded by $|\varphi|^2$. [9]

Now by using structural induction on the complexity of φ we will show that for every interval I and every subformula ξ of φ, $\mathcal{M} \models_I \xi$ implies $\mathcal{M}^* \models_I \xi$.

Base Case **:**
Suppose $\mathcal{M} \models_I \xi$

$\xi = \top$, $\xi = \bot$ or $\xi = \ell \sim k$: Trivial

$\xi = \int s \sim k$: For any $J \subseteq I$, $\langle J, s\rangle$ cannot be a redundant pair since there is no $J' \subseteq J$ or $J \subseteq J'$ such that $\langle J', s\rangle \in \mathcal{M}$. Therefore, $\langle J, s\rangle \in \mathcal{M}$ implies $\langle J, s\rangle \in \mathcal{M}^*$. Thus, $\mathcal{M}^* \models_I \int s \sim k$.

$\xi = \int_d s \leq k$: Similar to the above case.

Inductive Case**:**
Suppose $\mathcal{M} \models_I \xi$

$\xi = \langle e \sim k\rangle\psi$: By the semantics, there exists $\langle J, e\rangle \in \mathcal{M}$ such that $J \subseteq I$, $|J| \sim k$ and $\mathcal{M} \models_J \psi$. We choose such a J which is minimal under the order $\subseteq$, so that $\langle J, e\rangle \in \mathcal{M}^*$. By the inductive hypothesis, $\mathcal{M}^* \models_J \psi$. Thus, $\mathcal{M}^* \models_I \langle e \sim k\rangle\psi$.

$\xi = [e \sim k]\psi$: By the semantics, for every $J \subseteq I$, $\langle J, e\rangle \in \mathcal{M}$, and $|J| \sim k$ imply $\mathcal{M} \models_J \psi$. By construction, $\mathcal{M}^* \subseteq \mathcal{M}$. Since ξ is satisfied by $\mathcal{M}$, it has to be satisfied by its subset $\mathcal{M}^*$. By the inductive hypothesis, for every J, $\mathcal{M}^* \models_J \psi$. Thus, $\mathcal{M}^* \models_I [e \sim k]\psi$.

$\xi = \langle s \sim k\rangle^r_<\psi$: By the semantics, there exists $\langle J, s\rangle \in \mathcal{M}$ such that $R^r(J, I)$, $|J| \sim k$ and $\mathcal{M} \models_{fin(J,I)} \psi$. We know that $\langle J, s\rangle$ cannot be a redundant pair since there is no $J' \subset J$ or $J \subset J'$ such that $\langle J', s\rangle \in \mathcal{M}$. Therefore, $\langle J, s\rangle \in \mathcal{M}$ implies $\langle J, s\rangle \in \mathcal{M}^*$. By the inductive hypothesis, $\mathcal{M}^* \models_{fin(J,I)} \psi$. Thus, $\mathcal{M}^* \models_I \langle s \sim k\rangle^r_<\psi$.

$\xi = [s \sim k]^r_<\psi$: By the semantics, for every witness J, $\langle J, s\rangle \in \mathcal{M}$, $R^r(J, I)$ and $|J| \sim k$ imply $\mathcal{M} \models_{fin(J,I)} \psi$. We know that $\langle J, s\rangle$ cannot be a redundant pair. Therefore, for any witness J, $\langle J, s\rangle \in \mathcal{M}$ implies $\langle J, s\rangle \in \mathcal{M}^*$. By the inductive hypothesis, for every J, $\mathcal{M}^* \models_{fin(J,I)} \psi$. Then, $\mathcal{M}^* \models_{fin(J,I)} [s \sim k]^r_<\psi$.

$\xi = \langle s \sim k\rangle^r\psi$, $\xi = \langle s \sim k\rangle^r_>\psi$: Similar to $\xi = \langle s \sim k\rangle^r_<\psi$.

$\xi = [s \sim k]^r\psi$, $\xi = [s \sim k]^r_>\psi$: Similar to $\xi = [s \sim k]^r_<\psi$

□

APPENDIX B

In this section, we provide the technical details of the rules of the *expansion strategy*:

[9] A bound can also be given for the *width* of the model, but for the sake of simplicity we ignore it in the paper. We note that all the theorems and proofs can easily be extended to cover the width as well. Not surprisingly, the bound for the width will be exponential in the worst case because the formula might lead to exponential representation of the terms in the model.

Interval Relation Rule.
The *interval relation rule* guesses the interval relation between the given node and all other nodes in the graph. Let $\langle \mathcal{G}, \mathcal{C} \rangle$ be a tableau (where $\mathcal{G} = (V, E)$) and $v \in V$ be a node with $\lambda(v) = ([b_v, e_v], \rho(v), \mathcal{K}(v), \mathcal{L}(v), \bar{\mathcal{L}}(v))$. We remind that $\sim'$ negates the operator $\sim \in \{<, >, \leq, \geq, =\}$, e.g. the inverted operator of $<$ is $\geq$. The interval relation rule is defined as follows:

for any node $u \in V$ $(u \neq v)$

nondeterministically guess the interval relation between u and v:

v *before* u : Set $\mathcal{C} := \mathcal{C} \cup \{e_v < b_u\}$.

v *meets* u : Set $\mathcal{C} := \mathcal{C} \cup \{e_v = b_u\}$.

- **if** $\rho(v) \in \mathcal{S}$, $\rho(u) \in \mathcal{S}$ and $\rho(v) = \rho(u)$, **then** set $\mathcal{K}(v) := \{\bot\}$.

v *during* u : Set $\mathcal{C} := \mathcal{C} \cup \{b_v > b_u, e_v < e_u\}$, and expand E with a new edge from u to v (i.e. $E := E \cup \{u \to v\}$) if $\{v \to u\}$ has not been added previously (i.e. $v \to u \notin E$). In the sequel, this will be denoted as $\boldsymbol{E := E \cup \{u \to v\}}$). Do the following steps:

- **if** $\rho(v) \in \mathcal{S}$, $\rho(u) \in \mathcal{S}$ and $\rho(v) = \rho(u)$, **then** set $\mathcal{K}(v) := \{\bot\}$.
- **if** ($\rho(v) = e$ and $[e \sim k]\psi \in \mathcal{L}(u)$) or ($\rho(v) = s$ and $[s \sim k]^d\psi \in \mathcal{L}(u)$), **then** set either *i)* $\mathcal{C} := \mathcal{C} \cup \{(e_v - b_v) \sim' k\}$; or *ii)* $\mathcal{C} := \mathcal{C} \cup \{(e_v - b_v) \sim k\}$ and $\mathcal{K}(v) := \mathcal{K}(v) \cup \{\psi\}$.
- **if** $\rho(v) = s$ and $[s \sim k]^d_< \psi \in \mathcal{L}(u)$, **then** do either *i)* set $\mathcal{C} := \mathcal{C} \cup \{(e_v - b_v) \sim' k\}$; or *ii)* set $\mathcal{C} := \mathcal{C} \cup \{(e_v - b_v) \sim k\}$, add an immediate successor w with $\rho(w) = -$, $\mathcal{K}(w) = \{\psi\}$, $\mathcal{L}(w) = \emptyset$, $\bar{\mathcal{L}}(w) = \emptyset$, set $\mathcal{C} := \mathcal{C} \cup \{b_w = e_v, e_w = e_u\}$, $Q := Q \cup \{w\}$, $V := V \cup \{w\}$ and $E := E \cup \{u \to w, v \to w\}$.
- **if** $\rho(v) = s$ and $[s \sim k]^d_> \psi \in \mathcal{L}(u)$, **then** do the same steps above except set $\mathcal{C} := \mathcal{C} \cup \{b_w = b_u, e_w = b_v\}$.

v *equals* u : Set $\mathcal{C} := \mathcal{C} \cup \{b_v = b_u, e_v = e_u\}$, $E := E \cup \{u \to v\}$, and same steps as in the "during" case.

v *starts* u : Set $\mathcal{C} := \mathcal{C} \cup \{b_v = b_u, e_v < e_u\}$, $E := E \cup \{u \to v\}$, and same steps as in the "during" case.

v *finishes* u : Set $\mathcal{C} := \mathcal{C} \cup \{b_v > b_u, e_v = e_u\}$, $E := E \cup \{u \to v\}$, and same steps as in the "during" case.

v *overlaps* u : Set $\mathcal{C} := \mathcal{C} \cup \{b_v < b_u < e_v < e_u\}$, $E := E \cup \{u \to v\}$, and do the following:

- **if** $\rho(v) \in \mathcal{S}$, $\rho(u) \in \mathcal{S}$ and $\rho(v) = \rho(u)$, **then** set $\mathcal{K}(v) := \{\bot\}$.
- **if** $\rho(v) = s$ and $[s \sim k]^o\psi \in \mathcal{L}(u)$, **then** do either *i)* set $\mathcal{C} := \mathcal{C} \cup \{(e_v - b_v) \sim' k\}$; or *ii)* set $\mathcal{C} := \mathcal{C} \cup \{(e_v - b_v) \sim k\}$, add a successor w with $\rho(w) = -$, $\mathcal{K}(w) = \{\psi\}$, $\mathcal{L}(w) = \emptyset$, $\bar{\mathcal{L}}(w) = \emptyset$, set $\mathcal{C} := \mathcal{C} \cup \{b_w = b_u, e_w = e_v\}$, $Q := Q \cup \{w\}$, $V := V \cup \{w\}$, $E := E \cup \{v \to w, u \to w\}$.

- **if** $\rho(v) = s$ and $[s \sim k]^o_< \psi \in \mathcal{L}(u)$, **then** do the same step above except set $\mathcal{C} := \mathcal{C} \cup \{b_w = e_v, e_w = e_u\}$.
- **if** $\rho(v) = s$ and $[s \sim k]^o_> \psi \in \mathcal{L}(u)$, **then** do the same step above except set $\mathcal{C} := \mathcal{C} \cup \{b_w = b_v, e_w = b_u\}$.

The cases where v "after" u, v "met-by" u, v "overlapped-by" u, v "includes" u, v "started-by" u, and v "finished-by" u can be dealt with similarly. As seen above, each node has a label $\rho(v)$ which is either *root*, $-$, an event from $\mathcal{E}$ or a state from $\mathcal{S}$. Here, $-$ is used to label (dummy) nodes which represent intervals denoted by partial functions *init* and fin.

Universal Node Expansion Rule.
The *universal node expansion rule* expands all universal subformulas in $\bar{\mathcal{L}}(v)$. Let $\langle \mathcal{G}, \mathcal{C} \rangle$ be a tableau, and $v \in V$ be a node with $\lambda(v) = ([b_v, e_v], \rho(v), \mathcal{K}(v), \mathcal{L}(v), \bar{\mathcal{L}}(v))$. The *universal node expansion rule* for a node v is defined as follows:

for every $\xi \in \bar{\mathcal{L}}(v)$

if ξ=$[e \sim k]\psi$, **then for every** node $u \in V$ $(u \neq v)$ with $\rho(u) = e$ and u *non-strict-during* v, set either *i)* $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim' k\}$; or *ii)* $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim k\}$, $\mathcal{K}(u) := \mathcal{K}(u) \cup \{\psi\}$ and $Q := Q \cup \{u\}$.

if ξ=$[s \sim k]^r \psi$, **then for every** node $u \in V$ $(u \neq v)$ such that $\rho(u) = s$ and u R^r v, do either *i)* set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim' k\}$; or *ii)* set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim k\}$, and do the following:

- **if** $r \in \{d, e, s, f\}$, **then** set $\mathcal{K}(u) := \mathcal{K}(u) \cup \{\psi\}$ and $Q := Q \cup \{u\}$.
- **if** $r = o$, **then** add a successor w with $\rho(w) = -$, $\mathcal{K}(w) = \{\psi\}$, $\mathcal{L}(w) = \emptyset$, $\bar{\mathcal{L}}(w) = \emptyset$, and set $\mathcal{C} := \mathcal{C} \cup \{b_w = b_v, e_w = e_u\}$, $Q := Q \cup \{w\}$, $V := V \cup \{w\}$ and $E := E \cup \{v \to w, u \to w\}$.

if ξ=$[s \sim k]^r_< \psi$, **then for every** node $u \in V$ $(u \neq v)$ such that $\rho(u) = s$ and u R^r v, do either *i)* set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim' k\}$; or *ii)* set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim k\}$, and do the following:

- **if** $r \in \{d, e, s, f, o\}$, **then** add a successor w with $\rho(w) = -$, $\mathcal{K}(w) = \{\psi\}$, $\mathcal{L}(w) = \emptyset$, $\bar{\mathcal{L}}(w) = \emptyset$, set $\mathcal{C} := \mathcal{C} \cup \{b_w = e_u, e_w = e_v\}$, $Q := Q \cup \{w\}$, $V := V \cup \{w\}$ and $E := E \cup \{v \to w, u \to w\}$.

where u R^r v is true if the corresponding end-point constraints are in $\mathcal{C}$. For example, u R^d v is true if $b_u > b_v, e_u < e_v \in \mathcal{C}$. u "non-strict-during" v is true if u R^{desf} v is true. We can consider the cases where $r \in \{\overline{d}, \overline{s}, \overline{f}, \overline{o}\}$ in the same way. Also, ξ=$[s \sim k]^r_> \psi$ can be dealt with similarly.

Existential Node Expansion Rule.
The *existential node expansion rule* expands all existential subformulas in $\bar{\mathcal{L}}(v)$. Let $\langle \mathcal{G}, \mathcal{C} \rangle$ be a tableau (where $\mathcal{G} = (V, E)$), and $v \in V$ be a node with $\lambda(v) = ([b_v, e_v], \rho(v), \mathcal{K}(v), \mathcal{L}(v), \bar{\mathcal{L}}(v))$. The *existential node expansion rule* for a node v is defined as follows:

for every $\xi \in \bar{\mathcal{L}}(v)$

if $\xi = \langle e \sim k \rangle \psi$, **then** add an immediate successor w with $\rho(w) = e$, $\mathcal{K}(w) = \{\psi\}$, $\mathcal{L}(w) = \emptyset$, $\bar{\mathcal{L}}(w) = \emptyset$, and set $\mathcal{C} := \mathcal{C} \cup \{b_w \geq b_v, e_w \leq e_v, (e_w - b_w) \sim k\}$, $Q := Q \cup \{w\}$, $V := V \cup \{w\}$, $E := E \cup \{v \to w\}$.

if $\xi = \langle s \sim k \rangle^r \psi$, **then** add an immediate successor w with $\rho(w) = s$, $\mathcal{K}(w) = \emptyset$, $\mathcal{L}(w) = \emptyset$, $\bar{\mathcal{L}}(w) = \emptyset$, and set $\mathcal{C} := \mathcal{C} \cup \{(e_w - b_w) \sim k\}$, $Q := Q \cup \{w\}$, $V := V \cup \{w\}$, $E := E \cup \{v \to w\}$.

- **if** $r = d$, **then** set $\mathcal{C} := \mathcal{C} \cup \{b_w > b_v, e_w < e_v\}$ and $\mathcal{K}(w) := \{\psi\}$.
- **if** $r \in \{e, s, f\}$, **then** set $\mathcal{C}$ similar to the case $r = d$.
- **if** $r = o$, **then** set $\mathcal{C} := \mathcal{C} \cup \{b_w < b_v < e_w < e_v\}$, add a successor w' with $\rho(w') = -$, $\mathcal{K}(w') = \{\psi\}$, $\mathcal{L}(w') = \emptyset$, $\bar{\mathcal{L}}(w') = \emptyset$, and set $\mathcal{C} := \mathcal{C} \cup \{b_{w'} = b_v, e_{w'} = e_w\}$, $Q := Q \cup \{w'\}$, $V := V \cup \{w'\}$ and $E := E \cup \{v \to w', w \to w'\}$.

if $\xi = \langle s \sim k \rangle^r_< \psi$, **then** add two successors w, w' with $\rho(w) = s$, $\mathcal{K}(w) = \emptyset$, $\mathcal{L}(w) = \emptyset$, $\bar{\mathcal{L}}(w) = \emptyset$ and $\rho(w') = -$, $\mathcal{K}(w') = \{\psi\}$, $\mathcal{L}(w') = \emptyset$, $\bar{\mathcal{L}}(w') = \emptyset$, and set $\mathcal{C} := \mathcal{C} \cup \{b_{w'} = e_w, e_{w'} = e_v, (e_w - b_w) \sim k\}$, $Q := Q \cup \{w, w'\}$, $V := V \cup \{w, w'\}$ and $E := E \cup \{v \to w, v \to w', w \to w'\}$.

- **if** $r = d$, **then** set $\mathcal{C} := \mathcal{C} \cup \{b_w > b_v, e_w < e_v\}$.
- **if** $r = \{e, s, f, o\}$, **then** set $\mathcal{C}$ similar to the case $r = d$.

We can consider the cases where $r \in \{\overline{d}, \overline{s}, \overline{f}, \overline{o}\}$ in the same way. $\xi = \langle s \sim k \rangle^r_> \psi$ can be dealt with similarly.

Accumulation Rule.
The *accumulation rule* checks whether the total duration of a state satisfies a certain condition. It is defined as follows:

for every node $v \in V$ and $\xi \in \mathcal{L}'(v)$

if $\xi = \int s \sim k$, **then** let Sum be a temporary variable with $Sum := 0$,

for every node $w \in V$ with $\rho(w) = s$, do either

– set either $\mathcal{C} := \mathcal{C} \cup \{e_w \leq b_v\}$ or $\mathcal{C} := \mathcal{C} \cup \{b_w \geq e_v\}$ (i.e. $[b_w, e_w]$ either "before", "after", "meets" or "met-by" $[b_v, e_v]$);

– set either $\mathcal{C} := \mathcal{C} \cup \{b_w > b_v, e_w < e_v\}$ ($[b_w, e_w]$ "during" $[b_v, e_v]$), $\mathcal{C} := \mathcal{C} \cup \{b_w = b_v, e_w < e_v\}$ ($[b_w, e_w]$ "starts" $[b_v, e_v]$), $\mathcal{C} := \mathcal{C} \cup \{b_w > b_v, e_w = b_v\}$ ($[b_w, e_w]$ "finishes" $[b_v, e_v]$), or $\mathcal{C} := \mathcal{C} \cup \{b_w < b_v < e_w < b_v\}$ ($[b_w, e_w]$ "overlaps" $[b_v, e_v]$); and
set $Sum := Sum + min(e_v, e_w) - max(b_v, b_w)$.

Set $\mathcal{C} := \mathcal{C} \cup \{Sum \sim k\}$.

if $\xi = \int_d s \leq k$, **then** let Sum be a temporary variable with $Sum := 0$,

for every node $w \in V$ with $\rho(w) = s$, do either

– set either $\mathcal{C} := \mathcal{C} \cup \{e_w \leq b_v\}$ or $\mathcal{C} := \mathcal{C} \cup \{b_w \geq e_v\}$ (i.e. $[b_w, e_w]$ either "before", "after", "meets" or "met-by" $[b_v, e_v]$);

– set either $\mathcal{C} := \mathcal{C} \cup \{b_w > b_v, e_w < e_v\}$ ($[b_w, e_w]$ "during" $[b_v, e_v]$), $\mathcal{C} := \mathcal{C} \cup \{b_w = b_v, e_w < e_v\}$ ($[b_w, e_w]$ "starts" $[b_v, e_v]$), $\mathcal{C} := \mathcal{C} \cup \{b_w > b_v, e_w = b_v\}$ ($[b_w, e_w]$ "finishes" $[b_v, e_v]$), or $\mathcal{C} := \mathcal{C} \cup \{b_w < b_v < e_w < b_v\}$ ($[b_w, e_w]$ "overlaps" $[b_v, e_v]$);

for every node $w' \in V$ with $\rho(w') = s$, do either

- set either $\mathcal{C} := \mathcal{C} \cup \{e_{w'} \leq b_w\}$ or $\mathcal{C} := \mathcal{C} \cup \{b_{w'} \geq b_w + d$ (i.e. $[b_{w'}, e_{w'}]$ either "before", "after", "meets" or "met-by" $[b_w, b_w + d]$);
- set either $\mathcal{C} := \mathcal{C} \cup \{b_{w'} > b_w, e_{w'} < b_w + d\}$ ($[b_{w'}, e_{w'}]$ "during" $[b_w, b_w + d]$), $\mathcal{C} := \mathcal{C} \cup \{b_{w'} = b_w, e_{w'} < b_w + d\}$ ($[b_{w'}, e_{w'}]$ "starts" $[b_w, b_w + d]$), $\mathcal{C} := \mathcal{C} \cup \{b_{w'} > b_w, e_{w'} = b_w + d\}$ ($[b_{w'}, e_{w'}]$ "finishes" $[b_w, b_w + d]$), or $\mathcal{C} := \mathcal{C} \cup \{b_{w'} < b_w < e_{w'} < b_w + d\}$ ($[b_{w'}, e_{w'}]$ "overlaps" $[b_w, b_w + d]$); and
  set $Sum := Sum + min(b_w + d, e_{w'}) - max(b_w, b_{w'})$.

Set $\mathcal{C} := \mathcal{C} \cup \{Sum \leq k\}$.

In the accumulation rule we guess the interval relation of any node w with $\rho(w) = s$. If this relation is either "before", "after", "meets" or "met-by", then this node is not considered while calculating the total duration. Otherwise, we count it in the total duration.

Duration Rule.
The *duration rule* checks whether the length of the interval represented by the given node satisfies a certain condition. Let $\langle \mathcal{G}, \mathcal{C} \rangle$ be a tableau (where $\mathcal{G} = (V, E)$), and $v \in V$ be a node with $\lambda(v) = ([b_v, e_v], \rho(v), \mathcal{K}(v), \mathcal{L}(v), \bar{\mathcal{L}}(v))$. The duration rule is defined as follows:

for every $\xi \in \mathcal{L}'(v)$

if $\xi = \ell \sim k$, **then** set $\mathcal{C} := \mathcal{C} \cup \{(e_v - b_v) \sim k\}$.

APPENDIX C

Theorem 6.1. *The tableau method for ESDL terminates.*

Proof. Let $\langle \mathcal{G}, \mathcal{C} \rangle$ be a tableau constructed by the tableau procedure for a given a formula φ. The stopping condition ensures that infinite paths cannot occur in the tableau, and the depth of the longest path in the tableau is bounded by Lemma 4.2. Also, Rules 1, 2, 3, 4, 5 and 6 in the tableau procedure are applied finitely, i.e. the number of calls in these rule are bounded. Namely,

- *Rule 1*: bounded by the number of nodes existing in the tableau at the time of applying the rule;
- *Rule 2*: chooses only a disjunct from DNF;
- *Rule 3*: bounded by the number of subformulas and the number of existing nodes in the tableau;
- *Rule 4*: bounded by the number of subformulas;
- *Rule 5*: bounded by the number of subformulas and the number of existing nodes in the tableau at the time of applying the rule;
- *Rule 4*: bounded by the number of subformulas.

We also know that every node of $\mathcal{G}$ has a finite outgoing degree, because in the node expansion strategy we non-deterministically choose only one disjunct. Therefore, the tableau method terminates. □

Theorem 6.2. *Let φ be an ESDL formula in the normalized form. φ is satisfiable iff there is an open tableau for φ.*

Proof. **Soundness** ($\Leftarrow$) **:**
Suppose $\langle \mathcal{G}, \mathcal{C} \rangle$ (where $\mathcal{G} = (V, E)$) is an open tableau for φ. We pick some solution $\sigma : \mathcal{V} \mapsto \mathbb{R}$, which assigns real values to constraint variables in $\mathcal{C}$. Let $J_v = [\sigma(b_v), \sigma(e_v)]$ be the interval represented by node $v \in V$. We construct a model $\mathcal{M}$ as follows: $\mathcal{M} = \{\langle J_v, \rho(v) \rangle \mid \text{for any } v \in V \text{ s.t. } \rho(v) \notin \{root, -\}\}$.

Now we show that $\mathcal{M} \models_{I_0} \varphi$ (where I_0 is the initial interval). We claim that for every $v \in V$, $\mathcal{M} \models_{J_v} \mathcal{L}(v)$. We show, by structural induction, that $\phi \in \mathcal{L}(v)$ implies $\mathcal{M} \models_{J_v} \phi$. Note that, by construction of the tableau, $\mathcal{L}(v)$ comprises formulas of the forms $\top$, $\bot$, $\ell \sim k$, $\int s \sim k$, $\int_d s \leq k$, $\langle e \sim k \rangle \psi$, $[e \sim k]\psi$, $\langle s \sim k \rangle^r \psi$, $[s \sim k]^r \psi$, $\langle s \sim k \rangle^r_< \psi$, $[s \sim k]^r_< \psi$, $\langle s \sim k \rangle^r_> \psi$ and $[s \sim k]^r_> \psi$.

Base Case:

$\phi = \top$: Trivial

$\phi = \bot$: Since $\langle \mathcal{G}, \mathcal{C} \rangle$ is an open tableau, $\bot \notin \mathcal{L}(v)$.

$\phi = \ell \sim k$: By the duration rule, $\mathcal{C}$ contains $\{(e_v - b_v) \sim k\}$. So, we have $|J_v| \sim k$. Thus, $\mathcal{M} \models_{J_v} \phi$.

$\phi = \int s \sim k$: Let $\langle J_1, s \rangle, ..., \langle J_n, s \rangle \in \mathcal{M}$ $(n \geq 1)$ be all elements including s. By construction of the tableau, there exist nodes $u_1, ..., u_n$ such that

$J_{u_1} = J_1$, ... ,$J_{u_n} = J_n$. Assume that $J_1 R^i J_v$, ..., $J_n R^i J_v$ for $1 \leq i \leq n$ ($\mathcal{C}$, therefore, contains the corresponding constraints.) By the accumulation rule, $\mathcal{C}$ contains $[min(e_{u_1}, e_v) - max(b_{u_1}, b_v) + ... + min(e_{u_n}, e_v) - max(b_{u_n}, b_v)] \sim k$. This guarantees that $(|J_1 \cap J_v| + ... + |J_n \cap J_v|) \sim k$. Thus, $\mathcal{M} \models_{J_v} \phi$.

$\phi = \int_d s \leq k$: Similar to the case above.

***Inductive Case*:**

$\phi = \langle e \sim k \rangle \psi$: By the existential node expansion rule, there exists a node w with $\rho(w) = e$ and $\mathcal{K}(w) = \{\psi\}$. In addition, $\mathcal{C}$ contains $b_w \geq b_v, e_w \leq e_v$ and $(e_w - b_w) \sim k$. Let ψ be $\psi_1 \vee ... \vee \psi_n$ where $\psi_i = \psi_{i1} \wedge ... \wedge \psi_{in_i}$ ($n \geq 1$, $1 \leq i \leq n$ and $n_i \geq 1$). By Rule 2, $\psi_{i1}, .., \psi_{in_i} \in \mathcal{L}(w)$ for some i ($1 \leq i \leq n$). By the inductive hypothesis, $\mathcal{M} \models_{J_w} \psi_{i1} \wedge ... \wedge \mathcal{M} \models_{J_w} \psi_{in_i}$. Therefore, $\mathcal{M} \models_{J_w} \psi$. By construction of the tableau, we have $\langle J_w, e \rangle \in \mathcal{M}$ with $|J_w| \sim k$ and $J_w \subseteq J_v$. Thus, $\mathcal{M} \models_{J_v} \phi$.

$\phi = [e \sim k]\psi$: By the construction of $\mathcal{M}$, for any $J \in \mathcal{I}$ if $\langle J, e \rangle \in \mathcal{M}$, then there exists a node $u \in V$ such that $J_u = J$. According to the universal node expansion rule (or the interval relation rule) if $J_u \subseteq J_v$, then we do either: *i*) set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim' k\}$ ($\sim'$ is the corresponding inverted operator of $\sim$); or *ii*) set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim k\}$ and $\mathcal{K}(u) := \mathcal{K}(u) \cup \{\psi\}$.

Assume $|J_u| \sim k$ is false. Whatever the choice is, it is trivial to see that for any witness $J_u \subseteq J_v$, $\langle J_u, e \rangle$ and $|J_u| \sim k$ imply $\mathcal{M} \models_{J_u} \psi$. Assume $|J_u| \sim k$ is true. In this case, option i mentioned above cannot have been selected. Otherwise, $\mathcal{C}$ would contain $\{(e_u - b_u) \sim' k\}$, and it would result in an inconsistency. So option *ii* has been taken. In this case, we set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim k\}$ and $\mathcal{K}(u) := \mathcal{K}(u) \cup \{\psi\}$. Let ψ be $\psi_1 \vee ... \vee \psi_n$ where $\psi_i = \psi_{i1} \wedge ... \wedge \psi_{in_i}$ ($n \geq 1$, $1 \leq i \leq n$ and $n_i \geq 1$). By Rule 2, $\psi_{i1}, .., \psi_{in_i} \in \mathcal{L}(w)$ for some i ($1 \leq i \leq n$). By the inductive hypothesis, $\mathcal{M} \models_{J_u} \psi$. By construction of the tableau, we have $\langle J_u, e \rangle \in \mathcal{M}$. We also know that $J_u \subseteq J_v$ and $|J_u| \sim k$. Therefore, for any witness $J_u \subseteq J_v$, $\langle J_u, e \rangle$ and $|J_u| \sim k$ imply $\mathcal{M} \models_{J_u} \psi$. Thus, $\mathcal{M} \models_{J_v} \phi$.

$\phi = \langle s \sim k \rangle^r_< \psi$: By the existential node expansion rule, there exists a node w with $\rho(w) = s$. The existential rule extends $\mathcal{C}$ with $(e_w - b_w) \sim k$ and the corresponding constraints involving the enpoints b_w, e_w, b_v and e_v (These constraints are implied by R^r). The tableau algorithm creates a new node w' with $\rho(w') = -$ and $\mathcal{K}(w') = \{\psi\}$. Let $J_{w'}$ be the interval represented by the node w'. Since we extended $\mathcal{C}$ with some constraints involving the enpoints $b_w, e_w, b_{w'}, e_{w'}, b_v, e_v$ (These constraints are implied by $fin(J_w, J_v)$, we know that $J_{w'} = fin(J_w, J_v)$. By the inductive hypothesis and Rule 2, $\mathcal{M} \models_{J_{w'}} \psi$. By construction of the tableau, we have $\langle J_w, s \rangle \in \mathcal{M}$ with $|J_w| \sim k$ and $R^r(J_w, J_v)$. Thus, $\mathcal{M} \models_{J_v} \phi$.

$\phi = [s \sim k]^r_< \psi$: By the construction of $\mathcal{M}$, for any $J \in \mathcal{I}$ if $\langle J, s \rangle \in \mathcal{M}$, then there exists a node $u \in V$ such that $J_u = J$. According to the universal node expansion rule (or the interval relation rule) if $R^r(J_u, J_v)$, then we do either *i*) set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim' k\}$ ($\sim'$ is the corresponding inverted operator of $\sim$); or *ii*) set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim k\}$.

If $|J_u| \sim k$ is false, then trivially for any witness J_u, $\langle J_u, s \rangle, R^r(J_u, J_v)$ and $|J_u| \sim k$ imply $\mathcal{M} \models_{fin(J_u, J_v)} \psi$. If $|J_u| \sim k$ is true, then option i cannot

have been selected. Otherwise, $\mathcal{C}$ would contain $\{(e_u - b_u) \sim' k\}$, and it would result in an inconsistency. So option *ii* has been taken. In this case, we set $\mathcal{C} := \mathcal{C} \cup \{(e_u - b_u) \sim k\}$. The tableau algorithm creates a new node w with $\rho(w) = -$ and $\mathcal{K}(w) = \{\psi\}$. By the inductive hypothesis and Rule 2, $\mathcal{M} \models_{J_w} \psi$, where $J_w = fin(J_u, J_v)$. By construction of the tableau, we have $\langle J_u, s\rangle \in \mathcal{M}$. We also know that $R^r(J_u, J_v)$ and $|J_u| \sim k$. Thus, for any witness J_u, $\langle J_u, s\rangle, R^r(J_u, J_v)$ and $|J_u| \sim k$ imply $\mathcal{M} \models_{fin(J_u,J_v)} \psi$. Therefore, for either case $\mathcal{M} \models_{J_v} \phi$.

The other cases, where ϕ is either $\langle s \sim k\rangle^r \psi$, $[s \sim k]^r \psi$, $\langle s \sim k\rangle^r_> \psi$ or $[s \sim k]^r_> \psi$, can be dealt with similarly.

We have proved that for every $v \in V$, $\mathcal{M} \models_{J_v} \mathcal{L}(v)$. In particular, $\mathcal{M} \models_{I_0} \mathcal{L}(v_0)$. We know that $\mathcal{K}(v_0) = \{\varphi\}$. Now assume $\varphi = \varphi_1 \vee ... \vee \varphi_n$, where $\varphi_i = \varphi_{i1} \wedge ... \wedge \varphi_{in_i}$ ($n \geq 1$, $1 \leq i \leq n$ and $n_i \geq 1$). According to Rule 2 of the expansion strategy $\mathcal{L}(v_0) = \{\varphi_{i1}, ..., \varphi_{in_i}\}$ for some value of i. Therefore, we can easily conclude that $\mathcal{M} \models_{I_0} \varphi$.

Completeness ($\Rightarrow$) **:**
Suppose $\mathcal{M} \models_{I_0} \varphi$. By Lemma 4.2 there exists a model $\mathcal{M}^* \subseteq \mathcal{M}$, with depth at most $|\varphi|^2$, such that $\mathcal{M}^* \models_{I_0} \varphi$. We will show that there is an open tableau $\langle \mathcal{G}, \mathcal{C}\rangle$ for φ.

The *initial tableau* for φ is the tuple $\langle (v_0, \emptyset), \mathcal{C}_0\rangle$, where v_0 is the initial graph such that $\mathcal{K}(v_0) = \{\varphi\}$ and $\mathcal{L}(v_0) = \emptyset$, and $\mathcal{C}_0$ is the initial set of temporal constraints such that $\mathcal{C}_0 = \{b_{v_0} = beg(I_0), e_{v_0} = end(I_0), b_{v_0} \leq e_{v_0}\}$. A tableau $\langle \mathcal{G}, \mathcal{C}\rangle$ for φ is obtained by expanding the initial node v_0 through successive applications of the expansion strategy to existing nodes until no node remains to process, and by expanding the initial constraint set $\mathcal{C}_0$ with temporal constraints in the existing nodes.

According to the expansion strategy we apply the interval relation rule to the node v_0 as $\mathcal{L}(v_0)$ is empty. But since there is only one node, $\mathcal{K}(v_0)$ does not get updated. Let the disjunctive normal form of $\mathcal{K}(v_0) = \{\varphi\}$ be $\varphi_1 \vee ... \vee \varphi_n$, where $\varphi_i = \varphi_{i1} \wedge ... \wedge \varphi_{in_i}$ ($n \geq 1$, $1 \leq i \leq n$ and $n_i \geq 1$). Since $\mathcal{M}^* \models_{I_0} \varphi$, $\mathcal{M}^* \models_{I_0} \varphi_i$ for at least one value of i. So in Rule 2 of the expansion strategy we pick this value of i, so that $\mathcal{L}(v_0) = \{\varphi_{i1}, ..., \varphi_{in_i}\}$.

Now, we claim that for each node $v \in V$, there exists an interval J_v such that $\mathcal{M}^* \models_{J_v} \mathcal{L}(v)$ (Once we pick a witness J_v, it remains assigned to the node v until the tableau procedure terminates.) We prove the claim by induction on the stage in tableau construction at which the node v was created.

Base case **:**
Above we have shown that $\mathcal{M}^* \models_{I_0} \varphi_i$ for some value of i, and $\mathcal{L}(v_0) = \{\varphi_{i1}, ..., \varphi_{in_i}\}$. So, it is trivial to see $\mathcal{M}^* \models_{I_0} \mathcal{L}(v_0)$.

Inductive case **:**

Case 1:
Let $w \in V$ be a node with $\rho(w) = e$. Then w must have been created by the existential node expansion rule applied to a node v of which w is successor node. After the node w has been created, we apply the expansion strategy to the node w. So we first apply the interval relation rule. Let us consider two cases:

i) Application of the interval relation rule adds no material to $\mathcal{L}(w)$: Assume $\mathcal{L}(w) = \{\psi_0\}$ where $\psi_0 = \psi_{01} \wedge ... \wedge \psi_{0n_0}$ $(n_0 \geq 1)$. In this case, $\mathcal{L}(v)$ must contain $\xi = \langle e \sim k \rangle \psi$, where ψ has the form $\psi_0 \vee ... \vee \psi_l$ $(l \geq 0)$. By the inductive hypothesis a witness J_v is defined such that $\mathcal{M}^* \models_{J_v} \mathcal{L}(v)$. Let J_w be a witness for w. Thus, $\mathcal{M}^* \models_{J_w} \psi$.

When the existential rule was applied to v, we set $\mathcal{K}(w) := \{\psi\}$ and $\mathcal{C} := \mathcal{C} \cup \{b_w \geq b_v, e_w \leq e_v, (e_w - b_w) \sim k\}$. According to Rule 2 of the expansion strategy we select some of the disjunct of ψ, and extend $\mathcal{L}(w)$ with this disjunct. It is clear that ψ_0 is the subformula which was selected. So, $\mathcal{M}^* \models_{J_w} \psi_0$. Hence, $\mathcal{M}^* \models_{J_w} \mathcal{L}(w)$.

ii) Application of the interval relation rule adds some material to $\mathcal{L}(w)$: Assume $\mathcal{L}(w) = \{\psi_0, \psi_1, ..., \psi_m\}$ where $\psi_i = \psi_{i1} \wedge ... \wedge \psi_{im_i}$ $(0 \leq i \leq m$ and $m_i \geq 1)$, ψ_0 has been added to $\mathcal{L}(w)$ by applying the existential rule in v, and $\psi_1, ..., \psi_m$ have been added to $\mathcal{L}(w)$ by applying the interval relation rule to the node w. Above we have shown that $\mathcal{M}^* \models_{J_w} \psi_0$.

According to the interval relation rule we guess the interval relation between w and any node in V. Assume for any $1 \leq j \leq m$ ψ_j has been added to $\mathcal{L}(w)$ as a result of guessing the interval relation between w and a node u_j. Since $\mathcal{K}(w)$, therefore $\mathcal{L}(w)$, has been updated, this relation must have been either "equals", "during", "starts" or "finishes". In each case, $\mathcal{L}(u_j)$ must contain $\xi = [e \sim k]\psi$, where ψ has the form $\psi_j \vee ... \vee \psi_{j+l}$ $(l \geq 0)$. By the inductive hypothesis we have picked a witness J_{u_j} such that $\mathcal{M}^* \models_{J_{u_j}} \mathcal{L}(u_j)$; thus $\mathcal{M}^* \models_{J_{u_j}} \xi$. We know that $J_w \subseteq J_{u_j}$ because in the interval rule we have guessed the relation between J_w and J_{u_j} as either "equals", "during", "starts" or "finishes" (As we can see in the interval rule, $\mathcal{C}$ has been updated according to the corresponding non-deterministic choice of the relation.) We also know that $|J_w| \sim k$ because we have selected the option *ii* in the interval relation rule, and set $\mathcal{C} := \mathcal{C} \cup \{(e_w - b_w) \sim k\}$ (Otherwise, $\mathcal{K}(w)$ could not have been updated). Therefore, $\mathcal{M}^* \models_{J_w} \psi$.

When the interval rule was applied to w, we set $\mathcal{K}(w) := \mathcal{K}(w) \cup \{\psi\}$. Since ψ_j was added to $\mathcal{L}(w)$ when Rule 2 was applied, $\mathcal{M}^* \models_{J_w} \psi_j$ for any $1 \leq j \leq m$. Hence, $\mathcal{M}^* \models_{J_w} \mathcal{L}(w)$.

So, we have shown that once a node w is created, and the expansion strategy is applied, it is true that $\mathcal{M}^* \models_{J_w} \mathcal{L}(w)$. However, when new nodes are added to V, $\mathcal{L}(w)$ might get updated through the application of the universal node expansion rule in these nodes. So, we must show that whenever new material is added to $\mathcal{L}(w)$, $\mathcal{M}^* \models_{J_w} \mathcal{L}(w)$ remains true.

Now, assume $\mathcal{L}(w) = \{\psi_0, ..., \psi_m, \psi_{m+1}, ..., \psi_{m+n}\}$ where $\psi_i = \psi_{i1} \wedge ... \wedge \psi_{in_i}$ $(0 \leq i \leq m+n$ and $n_i \geq 1)$, and $\psi_{m+1}, ..., \psi_{m+n}$ have been added to $\mathcal{L}(w)$ by applying the universal node expansion rule to some nodes in V. Above we have shown that $\mathcal{M}^* \models_{J_w} \{\psi_0, ..., \psi_m\}$. Assume for any $m+1 \leq k \leq m+n$, ψ_k has been added to $\mathcal{L}(w)$ by applying the universal node expansion rule to a node $u_k \in V$. In this case, $\mathcal{L}(u_k)$ must contain $\xi = [e \sim k]\psi$, where ψ has the form $\psi_k \vee ... \vee \psi_{k+l}$ $(l \geq 0)$. By the inductive hypothesis we have picked a witness J_{u_k} such that $\mathcal{M}^* \models_{J_{u_k}} \mathcal{L}(u_k)$; thus $\mathcal{M}^* \models_{J_{u_k}} \xi$. We know that $J_w \subseteq J_{u_k}$. We

also know that $|J_w| \sim k$ because we have selected the option ii of the universal rule, and set $\mathcal{C} := \mathcal{C} \cup \{(e_w - b_w) \sim k\}$ (Otherwise, $\mathcal{K}(w)$ could not have been updated.) Therefore, $\mathcal{M}^* \models_{J_w} \psi$.

When the universal rule was applied to u_k, we set $\mathcal{K}(w) := \mathcal{K}(w) \cup \{\psi\}$. It is clear that ψ_k was selected when Rule 2 was applied. So, for any $m+1 \leq k \leq m+n$, $\mathcal{M}^* \models_{J_w} \psi_k$. Hence, $\mathcal{M}^* \models_{J_w} \mathcal{L}(w)$.

Case 2:
Let $w \in V$ be a node with $\rho(w) = s$. This case can be dealt with similar to Case 1.

Case 3:
Let $w \in V$ be a node with $\rho(w) = -$. Assume $\mathcal{L}(w) = \{\psi_0\}$ where $\psi_0 = \psi_{01} \wedge ... \wedge \psi_{0n_0}$ $(n_0 \geq 1)$. Then, the dummy node w must have been created by either the existential node expansion rule, the interval relation rule, or the universal node expansion rule. If it has been created by the existential rule, then $\mathcal{L}(v)$ of a node v of which w is a successor node must contain $\xi = \langle s \sim k\rangle^r_< \psi$ or $\xi = \langle s \sim k\rangle^r_> \psi$. Otherwise, $\mathcal{L}(u)$ of a node u at which the interval relation rule or the universal rule has been applied contains $\xi = [s \sim k]^r_< \psi$ or $\xi = [s \sim k]^r_> \psi$. By the inductive hypothesis, a witness J_v (J_u) is defined such that $\mathcal{M}^* \models_{J_v} \mathcal{L}(v)$ $(\mathcal{M}^* \models_{J_u} \mathcal{L}(u))$. By construction of the tableau, there exists a node w' with $\rho(w') = s$. Let $J_{w'}$ be a witness representing the node w'. It is trivial to see that $R^r(J_{w'}, J_v)$ $(R^r(J_{w'}, J_u))$ and $|J_{w'}| \sim k$. Since $\mathcal{M}^* \models_{J_v} \xi$, we have $\mathcal{M}^* \models_{J_w} \psi$, where $J_w = init(J_{w'}, J_v)$ or $J_w = fin(J_{w'}, J_v)$ $(J_w = init(J_{w'}, J_u)$ or $J_w = fin(J_{w'}, J_u))$.

When any of these rules (existential rule, universal rule and interval rule) was applied, we set $\mathcal{K}(w) := \{\psi\}$. Suppose ψ have the form $\psi_0 \vee ... \vee \psi_l$ $(l \geq 0)$. Since ψ_0 is the selected disjunct of ψ in Rule 2, $\mathcal{M}^* \models_{J_w} \psi_0$. Hence, $\mathcal{M}^* \models_{J_w} \mathcal{L}(w)$.

Therefore, we have proved that for each node $v \in V$, there exists an interval J_v such that $\mathcal{M}^* \models_{J_v} \mathcal{L}(v)$.

As can be seen, $\mathcal{C}$ was updated when we applied the existential node expansion rule, universal node expansion rule and interval relation rule. Also, some extra material is added to $\mathcal{C}$ when the accumulation and duration rules are applied to a node v. As we have a witness J_v for each node v, we must have a solution for $\mathcal{C}$. Therefore, $\mathcal{C}$ must be satisfiable. Meanwhile, we know the depth of the model $\mathcal{M}^*$ is at most $|\varphi|^2$ by the assumption. Since for any node $v \in V$, $\mathcal{M}^* \models_{J_v} \mathcal{L}(v)$, $\perp$ cannot be contained in $\mathcal{L}(v)$. It follows that $\langle \mathcal{G}, \mathcal{C}\rangle$ is an open tableau. □

Theorem 6.3. *The satisfiability problem for ESDL is in NEXPTIME.*

Proof. In Theorem 6.1 we have shown that the proposed method terminates. Now, we analyse its computational complexity. We now give a bound on the size of any tableau for φ.

In any node $v \in V$ we convert $\mathcal{K}(v)$ into DNF, and in some cases conversion to DNF can lead to an exponential increase in formula size. However, in the node expansion strategy we non-deterministically choose only one disjunct.

Therefore, the out degree of any node is bounded by $|\varphi|$. We also know that the depth of the longest path in the tableau is bounded by $|\varphi|^2$ by Lemma 4.2. Thus, the size of the tableau is bounded by $|\varphi|^{|\varphi|^2} = 2^{|\varphi|^2 log_2 |\varphi|}$ (Even if we considered the out degree of any node is bounded by $2^{|\varphi|}$, the tableau would be bounded by $2^{|\varphi|^{|\varphi|^2}} = 2^{|\varphi|^3}$.) So, the tableau procedure builds a tableau of size $2^{p(|\varphi|)}$ for some fixed polynomial p.

We can say that if an ESDL formula φ is satisfiable, then the tableau procedure constructs a graph from which a satisfying model $\mathcal{M}$ can be extracted, of size bounded by $2^{p(|\varphi|)}$ for some fixed polynomial p. □