Legal Knowledge and Information Systems
A. Wyner and G. Casini (Eds.)

doi:10.3233/978-1-61499-838-9-33

Timed Contract Compliance Under Event Timing Uncertainty

María-Emilia CAMBRONERO [a] and Luis LLANA [b] and Gordon J. PACE [c,1]
[a] *Department of Computer Science, University of Castilla-La Mancha, Albacete, Spain*
[b] *Department of Computer Science and Computation, Complutense | University of Madrid, Spain*
[c] *Department of Computer Science, University of Malta*

Abstract. Despite that many real-life contracts include time constraints, for instance explicitly specifying deadlines by when to perform actions, or for how long certain behaviour is prohibited, the literature formalising such notions is surprisingly sparse. Furthermore, one of the major challenges is that compliance is typically computed with respect to timed event traces with event timestamps assumed to be perfect. In this paper we present an approach for evaluating compliance under the effect of imperfect timing information, giving a semantics to analyse contract violation likelihood.

Keywords. Contract compliance, Formal semantics, Real-time contracts, Fuzzy time

1. Introduction

Many real-life contracts include concrete time constraints, whether placing limits by when obligations have to be discharged e.g. *"Money is to be made available to the client with 48 hours of a request for redemption"*, or whether it identifies a time window during which an event is prohibited e.g. *"Once disabled, a user may not log in for 1 hour"* or even through temporal delays e.g. *"After accessing the service for 30 minutes, the user is obliged to pay within 5 minutes or lose the right to use the service further"*. A number of contract languages which allow for the description of such real-time matters have been proposed in the literature, including ones based on deontic logic e.g. [1,2] and automata e.g. [3]. However, one common feature of these formal approaches is that they handle compliance analysis in a *crisp* manner — for a given contract and a sequence of timed events (each carrying a timestamp), they enable the identification of whether or not that trace violates the contract, giving a *yes* or *no* answer.

Consider the contract clause which states that $C \stackrel{df}{=}$ *"Once disabled, a user may not log in for 1 hour"*, and the following event trace:

$$tr \stackrel{df}{=} \langle (login, 02h24m58s), (disable, 02h25m02s), (login, 03h25m01s) \rangle$$

Typically, the analysis would deduce that contract C has been violated by trace tr due to the second *login* event happening within less than an hour of the *disable* event. The major

[1] Corresponding Author: Gordon J. Pace; E-mail: gordon.pace@um.edu.mt.

issue with such an analysis is the difficulty of obtaining perfect timestamps, particularly if the sources of events may occur in different locations.

In this paper we outline our initial attempts at adapting techniques originally developed for real-time logics and calculi [13] to enable compliance analysis starting from traces with fuzzy timed events i.e. the timestamp for each event is *not* a single point in time, but a function over time which indicates the likelihood of the event having happened at that point in time.

2. Fuzzy Timed Event Traces

Trace-based semantics of contracts define whether for a particular contract a given trace of events violates that contract or not. Given an alphabet of event names which can be observed EVENT, such semantics typically take a trace ranging over EVENT^*. When contracts refer to real-time, the traces have to be augmented to have every event tagged by a timestamp indicating when it occurred. Taking time to range over the non-negative real numbers: $\text{TIME} \stackrel{df}{=} \mathbb{R}_0^+$, a real-time trace tr ranges over sequences of timed events $tr \in (\text{EVENT} \times \text{TIME})^*$ (with the assumption that time is monotonically increasing along the trace), or just using a set of timed events $tr \in 2^{\text{EVENT} \times \text{TIME}}$ (since the timestamps implicitly indicate the ordering).

In our case, the time stamps are no longer exact point values, and we assume that we can only give a likelihood of an event having happened at a particular time. Thus, rather that associating every observed event with a single value over TIME, we will use an approach from fuzzy logic, giving a time distribution, associating every time value with the likelihood of the event having happened at that point in time i.e. $\text{TIMEDISTRIBUTION} \stackrel{df}{=} \text{TIME} \rightarrow [0,1]$.

Definition 1 *A* fuzzy-timed observation *is a pair* $\langle a, T \rangle \in \text{FUZZYOBSERVATION}$*, where* $a \in \text{EVENT}$ *is the event name, and* $T \in \text{TIMEDISTRIBUTION}$ *is the timing distribution of that event. A* fuzzy-timed trace $tr \in \text{FUZZYTRACE}$ *is a pair* $\langle es, \Delta \rangle$*, consisting of (i)* $es \in \mathscr{M}(\text{FUZZYOBSERVATION})$*, a finite multiset of fuzzy-timed observations*[2]*; and (ii)* $\Delta \in \text{TIME}$*, the event horizon indicating that the events recorded are from the initial time window from time 0 and* Δ *(i.e. events happening beyond this time window are not recorded).*

It is worth noting that the imprecision inherent in the traces is limited to the timing of the events. We assume that the multiset of event names recorded is faithful with respect to what really happened i.e. (i) all events are recorded (no events are lost); (ii) events are not wrongly observed (event integrity); and (iii) no extraneous events are inserted (no spontaneous generation of event).

Also note, that if the fuzzy observation distributions are probabilistic ones, and independent of each other, then we can use a probabilistic approach (e.g. if we are given two fuzzy-timed observations $\langle a_1, T_1 \rangle$ and $\langle a_2, T_2 \rangle$, then the probability of both events hap-

[2]We use the notation $\mathscr{M}(X)$ to denote multisets with elements from X. Note that a sequence of fuzzy-timed observations cannot be used, since there is now no canonical ordering of events, and neither is a set of observations sufficient, since we may observe two events with the same name and with the same distribution function.

pening at time t would be $T_1(t) \times T_2(t)$). However, since this independence is not always easy to guarantee, we adopt a fuzzy logic approach and will combine likelihood values using generic binary operators $\otimes$ (the likelihood of the two observations to happen, a *triangular norm* [11]) and $\oplus$ (the likelihood of either of the two observations to happen, a *triangular conorm*). We will write $\prod$ and $\sum$ for the generalised versions of $\otimes$ and $\oplus$.

We will define a number of operations on fuzzy observations and traces to be used in the rest of the paper.

Definition 2 *We will write eventHorizon(tr) to refer to the event horizon of trace tr i.e. eventHorizon$(\langle es, \Delta \rangle) \stackrel{df}{=} \Delta$. Fuzzy-timed observations and fuzzy-timed traces can be shifted earlier in time using the* time shift *operator $\ll$:*

$$\langle a, T \rangle \ll \delta \stackrel{df}{=} \langle a, \lambda t.T(t \ominus \delta) \rangle$$
$$\langle es, \Delta \rangle \ll \delta \stackrel{df}{=} \langle \{e \ll \delta \mid e \in es\},\ \Delta \ominus \delta \rangle$$

where $x \ominus y \stackrel{df}{=} \max(x - y, 0)$.

We define the function occurances, which given an event and a fuzzy-timed trace, returns a multiset of all time distributions which may occur according to the given trace:

$$occurances_a(\langle es, \Delta \rangle) \stackrel{df}{=} \{T \in \text{TIMEDISTRIBUTION} \mid \langle a, T \rangle \in es\}$$

Finally, we define the likelihood that for a given fuzzy-timed trace es, event a has not happened in the initial time window $[0, \delta]$, written $absence_{es}(a, \delta)$ as follows:

$$absence_{tr}(a, \delta) \stackrel{df}{=} \prod_{T \in occurances_a(tr)} 1 - \int_0^{\delta} T(t)\, dt$$

3. A Timed-Contract Language

In order to define compliance and violation of fuzzy-timed traces, we will take a real-time deontic logic covering obligations, prohibitions, recursion and reparations to show how typical deontic operators can be given a trace semantics under imprecisely timed observations. The syntax of the real-time deontic logic is the following:

$$\mathscr{C} ::= \top \mid \bot \mid wait_\delta(\mathscr{C}) \mid \mathscr{O}_{\leq \text{TIME}}(\text{EVENT})(\mathscr{C}, \mathscr{C}) \mid \mathscr{F}_{\leq \text{TIME}}(\text{EVENT})(\mathscr{C}, \mathscr{C}) \mid \mu X.\mathscr{C} \mid X$$

The core of the calculus are obligations and prohibitions, written as $\mathscr{O}_{\leq \delta}(a)(C_1, C_2)$ and $\mathscr{F}_{\leq \delta}(a)(C_1, C_2)$ respectively. Obligation $\mathscr{O}_{\leq \delta}(a)(C_1, C_2)$ indicates that event a is to be performed before δ time units pass. If a is performed before the deadline, the continuation contract C_1 starts being enforced, but if a is not performed within δ time units, the reparation contract C_2 instead starts being enforced. Dually, prohibition $\mathscr{F}_{\leq \delta}(a)(C_1, C_2)$ indicates that event a is prohibited for the upcoming δ time units. If a occurs in this time frame, the reparation contract C_2 is triggered, but if it does not, then the continuation contract C_1 starts being enforced instead. The fact that we give both obligation and prohibition modalities a continuation and reparation, the two modalities are direct duals of each other: $\mathscr{F}_{\leq \delta}(a)(C_1, C_2)$ yields the same top-level violations (i.e. ignoring violations for which a reparation is defined) as $\mathscr{O}_{\leq \delta}(a)(C_2, C_1)$.

The contract $wait_\delta(C)$ acts like contract C, but starting after δ time units have elapsed. The base contracts $\top$ and $\bot$ are used to denote the contracts which are, respectively, immediately satisfied and violated. Finally, the μ operator is used to denote re-

cursion — such that the contract $\mu X.C$ will act like contract C except that every free instance of X in C will act like $\mu X.C$ itself.

Note that, for simplicity, all obligations and prohibitions have continuations and reparations, but if these are not desired, one can use the base contracts $\top$ and $\bot$. For example, to state that one is obliged to logout in 30 minutes, with no reparation or continuation, one would write: $\mathscr{O}_{\leq 30}(logout)(\top,\bot)$. In the rest of the paper, to avoid syntactic overload we will leave out the $\top$ and $\bot$ continuation and reparation e.g. simply writing $\mathscr{O}_{\leq 30}(logout)$.

We will now give a fuzzy-timed trace semantics to the timed contract logic. It is worth observing that when giving a trace semantics for crisp observations, one would typically define a (crisp) relation between traces and contracts such that a trace and contract are related if and only if the trace led to a violation of the contract. In contrast, in the case of fuzzy-timed traces, such a relation can only provide fuzzy information — i.e. we will have a functions $[\![C]\!]^{tr}_{vio}$ indicating the likelihood of fuzzy-timed trace *tr* violating contract C.

We can define the semantics of the timed contract logic with respect to a fuzzy-timed trace in this manner. As the base case for the semantics, we can assert that a trace with an event horizon of 0 cannot result in a violation. Trivially violated and satisfied contracts similarly given certain results (1 and 0 respectively), while a contract starting with a *wait* clause simply shifts the timestamps of the trace and analyses the resulting trace with the continuation of the contract. Obligation is the most complex operator, for which we have to separately compute whether the obliged action is performed or not, and combine with the continuation and reparation of the obligation. Prohibition is given a semantics in terms of obligation, while the semantics of recursion uses unrolling of the definition.

Definition 3 *The trace semantics of violation are defined in terms of the violation function* $[\![-]\!]^{-}_{vio} \in \mathscr{C} \times \textsc{FuzzyTrace} \to [0,1]$*, such that for a given contract C and fuzzy trace tr,* $[\![C]\!]^{tr}_{vio}$ *gives the likelihood of the the observations in trace tr violating contract C, and is defined as follows:*

If $eventHorizon(tr) = 0$*:*

$$[\![C]\!]^{tr}_{vio} \stackrel{df}{=} 0$$

Otherwise:

$$[\![\top]\!]^{tr}_{vio} \stackrel{df}{=} 0$$

$$[\![\bot]\!]^{tr}_{vio} \stackrel{df}{=} 1$$

$$[\![wait_\delta(C)]\!]^{tr}_{vio} \stackrel{df}{=} [\![C]\!]^{tr \ll \delta}_{vio}$$

$$[\![\mathscr{O}_{\leq\delta}(a)(C_1,C_2)]\!]^{tr}_{vio} \stackrel{df}{=} \sum_{T \in occurances_a(tr)} \int_0^\delta absence_{tr}(a,t) \otimes T(t) \otimes [\![C_1]\!]^{tr\setminus\{(a,T)\} \ll t}_{vio}\, dt \oplus absence_{tr}(a,\delta) \otimes [\![C_2]\!]^{tr \ll \delta}_{vio}$$

$$[\![\mathscr{F}_{\leq\delta}(a)(C_1,C_2)]\!]^{tr}_{vio} \stackrel{df}{=} [\![\mathscr{O}_{\leq\delta}(a)(C_1,C_2)]\!]^{tr}_{vio}$$

$$[\![\mu X.C]\!]^{tr}_{vio} \stackrel{df}{=} [\![C[X \setminus \mu X.C]]\!]^{tr}_{vio}$$

Provided that all uses or recursion are guarded (i.e. the recursion variable occurs after an obligation, prohibition or wait), the finite event horizon, and the finite size of

the timed observations recorded in the trace guarantee termination of recursion, thus ensuring that the semantics are well-defined.

4. Related Work

In the literature the use of fuzzy logic approaches for contracts are typically limited to analysing possible observational continuations, e.g. computing the likelihood of future failure given what has already been observed e.g. [10]. Even when encoded within the logic, most work deals with a discrete time model. For instance, Figeri *et al.* [4] present a temporal logic Fuzzy-time Temporal Logic (FTL), to express the temporal imprecision allowing to express vague temporal notions such as *soon*. Crespo *et al.* [13] present another work considering the extension of time constraints with fuzzy methods for timed automata, while Alur *et al.* [14], extend timed automata with perturbed clocks.

In practically all these works discussed, it is worth noting that the fuzziness is typically dealt with at the logic level — the specification language or logic allows for the expression of fuzzy notions of time. Our approach takes the dual view, and assigns fuzziness to the timing of the observations. In terms of expressivity, the two approach seem to be equally expressive. However, we believe that our approach is more appropriate in a deontic setting. For instance, consider trying to regulate a speed limit of 30km/h in a particular area. In order to enforce such a regulation, cameras are used, with imprecise timers. Because of this imprecision, the police may decide to prosecute only those who were observed driving at 40km/h or faster, since even taking into account the timing imprecision, it can be ascertained that the speed limit was exceeded. If, however, the cameras are replaced with more accurate ones, it may become viable to (fairly) prosecute those exceeding just 35km/h. If we were to take the approach that fuzzy timing should appear in the regulation itself, one would have to update the regulations whenever a camera is changed to a more (or less) accurate one. In contrast, with our approach, the regulation remains unchanged, *"You may not exceed 30km/h"*, but the uncertainty distributions in the observations allow the calculation of the probability or likelihood of an observed car to have actually been overspeeding.

5. Conclusions

We have proposed a fuzzy time trace semantics for violations of timed contracts. The approach allows the factoring in of imprecise measurements when recording timestamps of events (e.g. due to communication lag or due to unsynchronised clocks) while still allowing the calculation of the likelihood of a violation of the contract actually having taken place. In contrast to specification languages, where observational error is typically encoded with the property or specification, in a deontic setting, we would like to keep a canonical form of the regulating text, and factor in the error in the input trace. In practice, such semantics can be used, for instance, to regulate financial transactions, where the distributed nature of the interacting subsystems (account holder, receiver of funds, node logging events, etc.) means that precise timing of events is virtually impossible.

As it stands, the work has a number of limitations which we are currently addressing. On one hand, we would like to extend the timed deontic logic to include conjunction and

choice over contracts, thus widening its expressivity. Furthermore, the approach we have presented in this paper places no constraint on the form of the time-stamp distribution functions, resulting in the semantics being of limited practical use due to difficulty in computing them. We are, however, exploring limiting these functions (e.g. limiting time-stamp distribution functions to trapezoidal shaped ones), in order to be able to compute the results of the semantics automatically.

References

[1] Jan M. Broersen, Frank Dignum, Virginia Dignum, and John-Jules Ch. Meyer. Designing a deontic logic of deadlines. In *7th International Workshop on Deontic Logic in Computer Science, DEON 2004, Madeira, Portugal, May 26-28, 2004. Proceedings*, pages 43–56, 2004.

[2] María Emilia Cambronero, Luis Llana, and Gordon J. Pace. A timed contract-calculus. Technical Report CS-2017-02, Department of Computer Science, University of Malta, 2017.

[3] Enrique Martínez, María-Emilia Cambronero, Gregorio Díaz, and Gerardo Schneider. Timed automata semantics for visual e-contracts. In *Proceedings Fifth Workshop on Formal Languages and Analysis of Contract-Oriented Software, FLACOS 2011, Málaga, Spain, 22nd and 23rd September 2011*, 2011.

[4] Achille Frigeri, Liliana Pasquale, and Paola Spoletini. Fuzzy time in LTL. *CoRR*, abs/1203.6278, 2012.

[5] Martin Leucker and César Sánchez. *Regular Linear Temporal Logic*, pages 291–305. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[6] Joaquin Perez, Jaime Jimenez, Asier Rabanal, Armando Astarloa, and Jesús Lázaro. Ftl-cfree: A fuzzy real-time language for runtime verification. *IEEE Trans. Industrial Informatics*, 10(3):1670–1683, 2014.

[7] Barbara Pernici and Seyed Hossein Siadat. Selection of service adaptation strategies based on fuzzy logic. In *SERVICES*, pages 99–106, 2011.

[8] Anderson Francisco Talon and Edmundo Roberto Mauro Madeira. Comparison between light-weight and heavy-weight monitoring in a web services fuzzy architecture. *Procedia Computer Science*, 64 (Complete):862–869, 2015.

[9] Anderson Francisco Talon and Edmundo Roberto Mauro Madeira. Improvement of e-contracts accomplishments by self-adaptive fuzzy architecture. In *2015 IEEE International Conference on Services Computing, SCC 2015, New York City, USA, 2015*, pages 507–514, 2015.

[10] Anderson Francisco Talon and Edmundo Roberto Mauro Madeira. A fuzzy scheduling mechanism for a self-adaptive web services architecture. In *ICEIS 2017 - Proceedings of the 19th International Conference on Enterprise Information Systems, Volume 1, Porto, Portugal, April 26-29, 2017*, pages 529–536, 2017.

[11] E.P. Klement, Radko Mesiar and Endre Pap. Triangular Norms. Springer Netherlands, 2000.

[12] Anderson Francisco Talon, Edmundo Roberto Mauro Madeira, and Maria Beatriz Felgar de Toledo. Self-adaptive fuzzy architecture to predict and decrease e-contract violations. In *2014 Brazilian Conference on Intelligent Systems, BRACIS 2014, Sao Paulo, Brazil, October 18-22, 2014*, pages 294–299, 2014.

[13] F. Javier Crespo, Alberto de la Encina and Luis Llana, Fuzzy-Timed Automata, in the *Proceedings of Formal Techniques for Distributed Systems 2010,* Springer Berlin Heidelberg 2010.

[14] Rajeev Alur, Salvatore La Torre, and P. Madhusudan, Perturbed Timed Automata, in *Lecture Notes in Computer Science: Hybrid Systems: Computation and Control 3414,* 2005.

[15] Rajeev Alur, Salvatore La Torre, and P. Madhusudan. Perturbed timed automata. In *Hybrid Systems: Computation and Control, 8th International Workshop, HSCC 2005, Zurich, Switzerland, March 9-11, 2005, Proceedings*, pages 70–85, 2005.

[16] Alexandre Donzé and Oded Maler. Robust satisfaction of temporal logic over real-valued signals. In *Formal Modeling and Analysis of Timed Systems - 8th International Conference, FORMATS 2010, Klosterneuburg, Austria, September 8-10, 2010. Proceedings*, pages 92–106, 2010.

[17] Georgios E. Fainekos and George J. Pappas. Robustness of temporal logic specifications for continuous-time signals. *Theoretical Computer Science*, 410(42):4262 – 4291, 2009.