Axiomatizing Equivalences over Regular Monitors*

Luca Aceto[1,2], Antonis Achilleos[1], Elli Anastasiadi[1], Anna Ingólfsdóttir[1]

[1] Dept. of Computer Science, Reykjavik University, Iceland
[2] Gran Sasso Science Institute, L'Aquila, Italy
luca@ru.is, luca.aceto@gssi.it, antonios@ru.is, elli19@ru.is, annai@ru.is

Abstract

We study whether recursion-free regular monitors have finite equational axiomatizations with respect to two notions of equivalence, namely verdict and ω-verdict equivalence.

1 Introduction

Equational axiomatizations provide a purely syntactic description of the chosen notion of equivalence over processes and characterize the essence of a process semantics by means of a few revealing axioms. We consider a fragment of the regular monitors described and studied by Aceto et al. in, for instance, [1, 4]. Monitors are a key tool in the field of runtime verification [3], where they are used to check for system properties by analyzing execution traces generated by processes.

A monitor is an agent that observes the events occurring in a system as it progresses through time. Two monitors are verdict equivalent when they characterize exactly the same traces as successful and failure ones. Similarly they are ω-verdict equivalent when they characterize the same infinite traces as successful and failure ones. Our goal in this work is to study the equational theory of those relations. We give a ground complete axiomatization for both of those equivalences. We also study open equations, provide an (infinite) complete axiomatization for verdict equivalence and we argue that no finite one exists, even when the set of actions monitors can analyze is finite. Such negative results are common in the field of process algebra [2] although they usually occur for more expressive languages.

We present a suitable notion of *normal form* and use it to reduce the verdict equivalence problem for closed monitors to equality between normal forms. We also study the complexity of checking equivalence between two closed monitors and find it to be almost linear (off by a constant factor) in the size of the syntax tree of the monitors. This result is in contrast with the $coNP$-completeness for equality testing between star-free regular expressions [6].

Apart from their intrinsic theoretical interest, axiomatizations such as the ones we present can form the basis for tools for proving equivalences between monitors using theorem-proving techniques and identify valid laws of "monitor programming" in the sense of [5]. A complete axiomatization captures all the valid laws of programming in a model-independent way. Such laws can, for instance, be used as a set of rewrite rules to bring a monitor into an equivalent but better (for instance, more succinct or canonical) form. Since monitors are often synthesized automatically from specifications of monitorable properties, non-optimal representations are very likely to arise as a result of monitor-synthesis algorithms. The availability of a complete axiomatization of monitor equivalence indicates that, at least for monitors written in the languages being axiomatized modulo the chosen notion of equivalence, one can always synthesize "optimal" monitors.

*The work reported in this paper is supported by the projects Open Problems in the Equational Logic of Processes (OPEL) (grant 196050-051) and 'TheoFoMon: Theoretical Foundations for Monitorability' (grant 163406-051) of the Icelandic Research Fund. Acetos work was also partially supported by the Italian MIUR PRIN 2017FTXR7S project IT MATTERS 'Methods and Tools for Trustworthy Smart Systems'.

2 Background

Syntax of monitors Let *Act* be a set of visible actions, ranged over by a, b. Following Milner [7], we use $\tau \notin Act$ to denote an unobservable action. We will denote the set of infinite sequences over *Act* as Act^ω. As usual Act^* stands for the set of finite sequences over *Act*. Let *Var* be a countably infinite set of variables, ranged over by x, y, z.

The collection Mon_F of regular, recursion-free monitors is the set of terms generated by the following grammar:

$$m, n ::= \; v \quad | \quad a.m \quad | \quad m + n \quad | \quad x \qquad\qquad v ::= end \;\;|\;\; yes \;\;|\;\; no,$$

where $a \in Act$ and $x \in Var$. The terms *end*, *yes* and *no* are called *verdicts*. Closed monitors are those that do not contain any occurrences of variables. For each $\alpha \in Act \cup \{\tau\}$, we define the transition relation $\xrightarrow{\alpha} \subseteq Mon_F \times Mon_F$ as the least one that satisfies the following rules:

$$\frac{}{a.m \xrightarrow{a} m} \qquad \frac{m \xrightarrow{\alpha} m'}{m + n \xrightarrow{\alpha} m'} \qquad \frac{n \xrightarrow{\alpha} n'}{m + n \xrightarrow{\alpha} n'} \qquad \frac{}{v \xrightarrow{\alpha} v} \; .$$

For $s = a_1 a_2 \dots a_n \in Act^*$ and $n \geq 0$, we use $m \overset{s}{\Rightarrow} m'$ to mean that:

1. $m(\xrightarrow{\tau})^* m'$ if $s = \varepsilon$, where ε stands for the empty string,
2. $m \overset{\varepsilon}{\Rightarrow} m_1 \xrightarrow{a} m_2 \overset{\varepsilon}{\Rightarrow} m'$ for some m_1, m_2 if $s = a \in Act$ and
3. $m \overset{a}{\Rightarrow} m_1 \overset{s'}{\Rightarrow} m'$ for some m_1 if $s = a.s'$.

If $m \overset{s}{\Rightarrow} m'$ for some m', we call s a trace of m.

Verdict and ω-Verdict Equivalence Let m be a (closed) monitor. We define:

$$L_a(m) = \{s \in Act^* \mid m \overset{s}{\Rightarrow} yes\} \text{ and } L_r(m) = \{s \in Act^* \mid m \overset{s}{\Rightarrow} no\}.$$

Note that we allow for monitors that may both accept and reject some trace. This is necessary to maintain our monitors closed under +. Of course, in practice, one is interested in monitors that are consistent in their verdicts.

Definition 1. *Let m and n be closed monitors. We say that m and n are* ***verdict equivalent****, written $m \simeq n$, iff $L_a(m) = L_a(n)$ and $L_r(m) = L_r(n)$. We say that m and n are* ***ω-verdict equivalent****, written $m \simeq_\omega n$, iff $L_a(m) \cdot Act^\omega = L_a(n) \cdot Act^\omega$ and $L_r(m) \cdot Act^\omega = L_r(n) \cdot Act^\omega$. These equivalences are extended to open monitors in the standard way.*

Lemma 1. *The following hold: (1) $\simeq$ and $\simeq_\omega$ are both congruences. (2) $\simeq \;\subseteq\; \simeq_\omega$ and the inclusion is strict when Act is finite. (3) If Act is infinite then $\simeq \;=\; \simeq_\omega$.*

- $A1$: $x + y = y + x$
- $A2$: $x + (y + z) = (x + y) + z$
- $A3$: $x + x = x$
- $A4$: $x + end = x$
- $E1_a$: $a.end = end$ $(a \in Act)$
- Y_a: $yes = yes + a.yes$ $(a \in Act)$
- N_a: $no = no + a.no$ $(a \in Act)$
- $D1_a$: $a.(x + y) = a.x + a.y$ $(a \in Act)$

3 Results on axiomatizations and complexity

Our axiom system for verdict equivalence over closed monitors is $\mathcal{E}_{veq}$, whose axioms are:

Theorem 1. *$\mathcal{E}_{veq}$ is complete over closed monitors modulo $\simeq$. That is if m, n are closed monitors in Mon_F and $m \simeq n$ then $\mathcal{E}_{veq} \vdash m = n$. Moreover, $\mathcal{E}_{veq}$ is complete over closed terms modulo $\simeq_\omega$ when Act is infinite.*

In order to capture ω-verdict equivalence when Act is finite, we extend the axiom set by: $\mathcal{E}_{\omega-veq} = \mathcal{E}_{veq} \cup \{yes = \sum_{a \in Act} a.yes\} \cup \{no = \sum_{a \in Act} a.no\}$. We then prove:

Theorem 2. *$\mathcal{E}_{\omega-veq}$ is complete over closed terms modulo $\simeq_\omega$. That is if m, n are closed monitors in Mon_F and $m \simeq_\omega n$ then $\mathcal{E}_{\omega-veq} \vdash m = n$,when Act is finite.*

Naturally we continue towards the relevant results for open terms. Initially we only expand the axioms as: $\mathcal{E}'_{veq} = \mathcal{E}_{veq} \cup \{yes + no + x = yes + no\}$ and then prove:

Theorem 3. *$\mathcal{E}'_{veq}$ is complete for open terms modulo $\simeq$ when Act is infinite. That is, if m, n are open monitors in Mon_F and $m \simeq n$ then $\mathcal{E}'_{veq} \vdash m = n$.*

The final part of this work is dedicated to determining an axiomatization with respect to $\simeq$ and $\simeq_\omega$ for open equations when Act is finite and it includes an analysis on why the axiomatization cannot be finite.

Finally regarding the complexity of determining whether two monitors are equivalent, our completeness proof suggests that discovering a normal form for a monitor implies a somewhat pre-defined application of our axioms. In the case of closed terms the "normalization" procedure with respect to verdict equivalence can take place recursively and in a way mimicking the inductive proof of the existence of a normal form. After this is done then the equality testing of two monitors is trivially testing equality for two rooted, ordered and labelled trees. The final complexity of the algorithm is $O(n \cdot k \cdot log(k))$ where k is the size of Act and n is the sum of the sizes of the syntactic trees of the tested monitors. Future work includes the complexity analysis for verdict and ω-verdict equivalence testing between open monitors and the study of the equational theory of regular monitors with recursion.

References

[1] Luca Aceto, Antonis Achilleos, Adrian Francalanza, Anna Ingólfsdóttir, and Karoliina Lehtinen. Adventures in monitorability: From branching to linear time and back again. *Proc. ACM Program. Lang.*, 3(POPL):52:1–52:29, 2019.

[2] Jos C.M. Baeten, Twan Basten, and Michel A. Reniers. *Process Algebra: Equational Theories of Communicating Processes*, volume 50 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, 2009.

[3] Ezio Bartocci and Yliès Falcone, editors. *Lectures on Runtime Verification: Introductory and Advanced Topics*, volume 10457 of *Lecture Notes in Computer Science*. Springer, 2018.

[4] Adrian Francalanza, Luca Aceto, and Anna Ingólfsdóttir. Monitorability for the Hennessy–Milner Logic with recursion. *Form. Methods Syst. Des.*, 51(1):87–116, 2017.

[5] C. A. R. Hoare, Ian J. Hayes, Jifeng He, Carroll Morgan, A. W. Roscoe, Jeff W. Sanders, Ib Holm Sørensen, J. Michael Spivey, and Bernard Sufrin. Laws of programming. *Commun. ACM*, 30(8):672–686, 1987.

[6] Harry B. Hunt, Daniel J. Rosenkrantz, and Thomas G. Szymanski. On the equivalence, containment, and covering problems for the regular and context-free languages. *J. Comput. Syst. Sci.*, 12:222–268, 1976.

[7] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.