Axiomatizing recursion-free, regular monitors*

Luca Aceto[a,b], Antonis Achilleos[a], Elli Anastasiadi[a,*], Anna Ingolfsdottir[a]

[a] *ICE-TCS, Department of Computer Science, Reykjavik University, Iceland*
[b] *Gran Sasso Science Institute, L'Aquila, Italy*

Abstract

Monitors are a key tool in the field of runtime verification, where they are used to verify system properties by analyzing execution traces generated by processes. Work on runtime monitoring carried out in a series of papers by Aceto et al. has specified monitors using a variation on the regular fragment of Milner's CCS and studied two trace-based notions of equivalence over monitors, namely verdict and ω-verdict equivalence. This article is devoted to the study of the equational logic of monitors modulo those two notions of equivalence. It presents complete equational axiomatizations of verdict and ω-verdict equivalence for closed and open terms over recursion-free monitors. It is also shown that verdict equivalence has no finite equational axiomatization over open monitors when the set of actions is finite and contains at least two actions.

Keywords: Monitors, Formal Verification, CCS, Equational Logic, Processes, Process Algebra, Axiomatization, Trace Equivalence, Verdicts.

1. Introduction

The search for equational axiomatizations of a notion of equivalence over some process description language is one of the classic topics in concurrency theory, as witnessed by the literature on this subject over the last forty years. (See, for instance, [7, 8, 9, 16, 18, 27, 33, 34, 35, 43, 44] for early references as well as survey and textbook accounts, and the papers [4, 5, 28, 37] for examples of the rich body of recent contributions to this field.) This research avenue has its intellectual roots in the time-honored study of the existence of finite,

*This article is based on material presented at the 31st Nordic Workshop on Programming Theory, NWPT 2019, in Tallinn. The authors were supported by the projects 'Open Problems in the Equational Logic of Processes' (OPEL) (grant No 196050-051) and 'Mode(l)s of Verification and Monitorability' (MoVeMent) (grant No 217987) of the Icelandic Research Fund, and 'Runtime and Equational Verification of Concurrent Programs' (ReVoCoP) (grant No 222021), of the Reykjavik University Research Fund. Luca Aceto's work was also partially supported by the Italian MIUR PRIN 2017 project FTXR7S IT MATTERS 'Methods and Tools for Trustworthy Smart Systems'.

*Elli Anastasiadi, Menntavegur 1, 102 Reykjavik, Iceland

Email addresses: luca@ru.is, luca.aceto@gssi.it (Luca Aceto), antonios@ru.is (Antonis Achilleos), elli19@ru.is (Elli Anastasiadi), annai@ru.is (Anna Ingolfsdottir)

(conditional) equational proof systems for equality of regular expressions, as presented in [20, 38, 39, 48, 49].

There are manifold reasons for studying equational axiomatizations of equivalences over processes. For example, the existence of a finite, or at least finitely specified, equational axiomatization for some notion of process equivalence is often considered as one of the yardsticks to assess its mathematical tractability. Additionally, 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, which can be used to compare a variety of semantics in a model-independent way (as done, for instance, in [27]). Moreover, such axiomatizations pave the way to the use of theorem-proving techniques to establish that two process descriptions express the same behavior modulo the chosen notion of behavioral equivalence [21, 29, 41], and also play an important role in the partial evaluation of programs [32].

In this paper, we study the equational logic of the *monitors* studied by Aceto et al. in, for instance, [1, 2, 26]. Monitors are a key tool in the field of runtime verification (see [13, 24, 30, 31, 40, 46, 51, 52] and the references therein for an overview of this active research area), where they are used to check for system properties by analyzing execution traces generated by processes and are often expressed using some automata-based formalism. The notion of monitorable property has been defined in a seminal paper by Pnueli and Zaks [46]. Intuitively, a property of finite and infinite system executions is s-monitorable, for some finite trace of observable events s, if there is an extension of s after which a monitor will be able to determine conclusively whether the observed system execution satisfies or violates the property. This means that verdicts issued by monitors are *irrevocable*. In that work by Pnueli and Zaks, a property is described by the set of finite and infinite executions that satisfy it. However, in the theory and practice of runtime verification, one often specifies properties finitely using formalisms such as automata or (variations on) temporal logics and studies what specifications in the chosen formalism are 'monitorable' and with what correctness guarantees—see, for instance, [12, 15, 47]. Since monitors are part of the trusted computing base, the automated, correct-by-design *monitor synthesis* from the formal specification of properties has been thoroughly studied in the literature and is often accompanied by the experimental evaluation of the overhead induced by monitoring—see, for example, the study of various approaches to the automated monitor synthesis for systemC specifications given in [52] and the framework for benchmarking of runtime verification tools presented in [3].

In [1, 2, 26], Aceto et al. specified monitors using a variation on the regular fragment of Milner's CCS [42] and studied two trace-based notions of equivalence over monitors, namely verdict and ω-verdict equivalence. Intuitively, two monitor descriptions are verdict equivalent when they accept and reject the same finite execution traces of the systems they observe. The notion of ω-verdict equivalence is the 'asymptotic version' of verdict equivalence, in that it is solely concerned with the infinite traces that are accepted and rejected by

monitors. In their work, Aceto et. al. focus on determining the 'monitorable' fragment of Hennessy-Milner Logic with recursion [1, 26] and provide monitor-synthesis algorithms for properties that can be expressed in that fragment. The key (and non-negotiable) property that the monitor synthesized from a formula φ in the monitorable fragment of that logic should satisfy is *soundness*, which means that a verdict issued by the monitor as it examines a system execution determines whether that execution satisfies φ or not correctly. Naturally, sound monitors cannot produce contradictory verdicts for a given trace.

Our contribution. When monitors are described by expressions in some monitor-specification language, such as the one employed by Aceto et al. in *op. cit.*, it is natural to ask oneself whether one can (finitely) axiomatize notions of monitor equivalence over (fragments of) that language. This study is devoted to addressing that question in the simplest non-trivial setting. In particular, in order to stay within the realm of classic equational logic over total algebras, we consider a language that allows one to specify unsound monitors. However, all the results we present in the paper specialize to sub-languages consisting of (sound) monitors that can only issue either positive or negative verdicts.

The main results we present in this paper are complete equational characterizations of verdict equivalence over both closed (that is, variable-free) and open, recursion-free regular monitors. More specifically, we first provide an equational axiomatization of verdict equivalence over closed terms from the language of monitors we study that is finite if so is the set of actions monitors can observe (Theorem 2). The landscape of axiomatizability results for verdict equivalence over open terms turns out to be more varied. This variety is witnessed by the fact that there are three different axiomatizations, depending on whether the set of actions is infinite (Theorem 4), finite and containing at least two actions (Theorem 5) or a singleton (Theorem 6). Only the axiomatization given in Theorem 6 is finite and we show that this is unavoidable. Indeed, verdict equivalence has no finite equational basis when the set of actions is finite and of cardinality at least two (Theorem 10).

It turns out that the above-mentioned axiomatizations are also complete for ω-verdict equivalence if the set of actions that monitors may observe is infinite, as in that case the two notions of equivalence coincide. On the other hand, if the set of actions is finite, ω-verdict equivalence is strictly coarser than verdict equivalence. We also provide a finite, complete axiomatization of ω-verdict equivalence for closed monitors in the setting of a finite set of actions (Theorem 3). Our Theorem 8 gives a complete axiomatization of ω-verdict equivalence over open monitors when the set of actions contains at least two actions. If the set of actions is a singleton, ω-verdict equivalence has a finite equational basis (Theorem 7).

The equational axiomatizations we present in this article capture the 'laws of monitor programming' [35] for an admittedly rather inexpressive language. Indeed, recursion-free regular monitors describe essentially tree-like finite-state automata with distinguished accept and reject states at their 'leaves' with self-loops labeled by every action. (See the operational semantics of monitors in

Table 1. Note, however, that those automata may have infinitely many transitions, if the set of actions monitors can observe is infinite. As shown already by Milner in his classic books on CCS [42, 44], this feature is useful when modeling system events that carry data values. See, for instance, the paper [11] for one of the earliest attempts to incorporate data into runtime verification.) However, as witnessed by our results and their proofs, the study of the equational theory of monitors modulo the notions of equivalence we consider is non-trivial even for the minimal language studied in this paper. In our, admittedly biased, opinion, it is therefore worthwhile to map the territory of axiomatizability results for recursion-free regular monitors, since results for more expressive languages will have to build upon those we obtain in this article. We remark, in passing, that the non-finite axiomatizability result in Theorem 10 is obtained over a substantially more restrictive syntax than classic negative results for the algebra of regular expressions, which rely on the hardness of expressing the interplay between Kleene star and concatenation equationally [6, 20, 48].

The contribution of this paper is entirely theoretical and we make no claims pertaining to the applicability of our current results in the practice of runtime verification. However, apart from their intrinsic theoretical interest, (extensions of) the equational axiomatizations we present might be used in the automatic, syntax-driven synthesis of monitors from specifications of 'monitorable properties', as presented in [1, 2, 25], to rewrite monitor expressions in an 'equivalent, but simpler' syntactic form, for instance by eliminating 'redundant' subexpressions. As witnessed by the study of optimized temporal monitors for SystemC presented in [52], the investigation of monitor optimizations based on equational rewriting or other techniques requires a substantial experimental research effort and is outside the scope of this article. We discuss other avenues for future research in Section 6.

2. Preliminaries

We begin by introducing recursion-free regular monitors (or simply monitors in this study) and the two notions of verdict equivalence that we study in this paper. We refer the interested reader to [1, 26] for background motivation and more information.

Syntax of monitors. Let Act be a set of visible actions, ranged over by a, b. Following Milner [44], we use $\tau \notin Act$ to denote an unobservable action. The symbol α ranges over $Act \cup \{\tau\}$. Let Var be a countably infinite set of variables, ranged over by x, y, z. We assume that $Act \cup \{\tau\}$ and Var are disjoint.

We write Act^{ω} for the set of infinite sequences over Act. As usual, Act^{*} stands for the set of finite sequences over Act. Let A be a set of finite sequences and B be a set of sequences. We write $A \cdot B$ for the concatenation of A and B.

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

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

where $a \in Act$ and $x \in Var$. The terms *end*, *yes* and *no* are called *verdicts*. Intuitively, *yes* stands for the acceptance verdict, *no* denotes a rejection verdict and *end* is the inconclusive verdict, namely the state a monitor reaches when, based on the sequence of observations it has processed so far, it realizes that it will not be able to issue an acceptance or rejection verdict in the future. As will be formalized by the operational semantics of monitors to follow, verdicts are irrevocable. This means that once a monitor reaches a verdict, it will stick to it regardless of what further observations it makes. See, for instance, [1, 13, 26] for a detailed technical discussion.

Intuitively, a monitor of the form $a.m$ can observe action a and behave like m thereafter. On the other hand, a monitor of the form $m+n$ can behave either like m or like n.

Remark 1. *The work on which we build in this paper considers a setting with three verdicts, two of which are 'conclusive.' There are a number of other approaches in the field of runtime verification that consider many-valued verdicts. We refer the interested reader to, for instance, [10, 12, 14, 17, 23] for further information.*

Closed monitors are those that do not contain any occurrences of variables. A (closed) *substitution* is a mapping σ from variables to (closed) monitors. We write $\sigma(m)$ for the monitor that results when applying the substitution σ to m. Note that $\sigma(m)$ is closed, if σ is a closed substitution.

Definition 1 (**Notation**). *We use $m\ [+v]$ for a verdict v to indicate that v is an optional summand of m, that is, that the term can be either m or $m + v$. In addition a monitor will be called v-free for a verdict v, when it does not contain any occurrences of v.*

For a finite index set $I = \{i_1, \ldots, i_k\}$ and indexed set of monitors $\{m_i\}_{i \in I}$, we write $\sum_{i \in I} m_i$ to stand for end if $I = \emptyset$ and for $m_{i_1} + \ldots + m_{i_k}$ otherwise. This notation is justified by the fact that $+$ is associative and commutative, and has end as a neutral element, in all of the semantics we use in this paper.

We now associate a notion of syntactic depth with each monitor. Intuitively, the decision a monitor m takes when reading a string $s \in Act^*$ only depends on the prefixes of s whose length is at most the syntactic depth of m.

Definition 2 (**Syntactic Depth**). *For any closed monitor $m \in Mon_F$, we define $depth(m)$ as follows:*

- $depth(a.m) = 1 + depth(m)$,
- $depth(m_1 + m_2) = \max(depth(m_1), depth(m_2))$ *and*
- $depth(v) = 0$ *for a verdict v.*

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

Table 1: Operational semantics of processes in Mon_F.

Semantics of monitors. 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 axioms and rules in Table 1.

For example, $yes + x \xrightarrow{\tau} yes$ and $a.yes + end \xrightarrow{b} end$, for each $a, b \in Act$. A useful fact based on the above operational semantics is that if $m \xrightarrow{\tau} m'$, then $m' = v$ for some verdict v.

Note that variables have no transitions. They represent under-specification in monitor behavior. For instance, monitor $a.yes + x$ is one that we know can reach the verdict *yes* after having observed an a action. Further information on the behavior of that monitor can only be gleaned once the variable x has been instantiated via a (closed) substitution.

For m, m' in Mon_F and $s = a_1 \dots a_k$ in Act^*, $k \geq 0$, $m \xrightarrow{s} m'$ holds iff there are $m_0, , \dots, m_k$ such that

$$m = m_0 \xrightarrow{a_1} m_1 \cdots m_{k-1} \xrightarrow{a_k} m_k = m'.$$

Additionally, for $s \in Act^*$, we use $m \stackrel{s}{\Rightarrow} m'$ to mean that:

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

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

Lemma 1. *For all $s \in Act^*$, $m, n \in Mon_F$, and verdict v, $m + n \stackrel{s}{\Rightarrow} v$ iff $m \stackrel{s}{\Rightarrow} v$ or $n \stackrel{s}{\Rightarrow} v$.*

Proof. We prove both implications separately, by induction on the length of s. The details are straightforward and are therefore omitted. Here we limit ourselves to remarking that, in the proof of the implication from right to left, if $s = \varepsilon$ and $m = v$, say, then $v + n \xrightarrow{\tau} v$ by the rules in Table 1. □

Remark 2. *Note that the implication from right to left in Lemma 1 would not hold in the absence of rule $v \xrightarrow{\tau} v$ in Table 1.*

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

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

Intuitively, $L_a(m)$ denotes the set of traces that are accepted by m, whereas $L_r(m)$ stands for the set of traces that m rejects. The sets $L_a(m)$ and $L_r(m)$ will also be referred to as the acceptance and rejection set of m respectively. Note that we allow for monitors that may both accept and reject the same trace. This is necessary to maintain our monitors closed under $+$ and to work with classic total algebras rather than partial ones. Of course, in practice, one is interested in monitors that are consistent in their verdicts. One way to ensure consistency in monitor verdicts, which was considered in [26], is to restrict oneself to monitors that use only one of the conclusive verdicts *yes* and *no*. All the results that we present in the remainder of this paper apply to such monitors.

Remark 3. *The reader might wonder about the connection between the languages that are accepted/rejected by recursion-free regular monitors and star-free languages [50]. A simple argument by induction on the structure of monitors shows that every recursion-free regular monitor denotes a pair of star-free languages, one for its acceptance set and one for its rejection set. Moreover, this means that recursion-free regular monitors correspond to properties that can be expressed in LTL [36]. However, there are star-free languages (and therefore LTL properties) that cannot be described by recursion-free regular monitors. For example, the language $(ab)^*$ is star-free (see, for instance, [22, page 267]) but does not correspond to any recursion-free regular monitor.*

The monitors we consider in this paper output a positive or negative verdict after a finite number of computational steps, if they do so at all. This means that the linear-time temporal properties to which their acceptance and rejection set correspond are both 'Always Finitely Refutable' and 'Always Finitely Satisfiable' in the sense of [45], as proven in [1].

Definition 3. *Let m and n be closed monitors.*

- *We say that m and n are **verdict equivalent**, written $m \simeq n$, if $L_a(m) = L_a(n)$ and $L_r(m) = L_r(n)$.*
- *We say that m and n are **ω-verdict equivalent**, written $m \simeq_\omega n$, if $L_a(m) \cdot Act^\omega = L_a(n) \cdot Act^\omega$ and $L_r(m) \cdot Act^\omega = L_r(n) \cdot Act^\omega$.*

For open monitors m and n, we say that $m \simeq n$ if $\sigma(m) \simeq \sigma(n)$, for all closed substitutions σ. The relation $\simeq_\omega$ is extended to open monitors in similar fashion.

Example 1. *It is easy to see that $m + end \simeq m$ holds for each $m \in Mon_F$. Moreover, since $L_a(end) = \emptyset$ and $L_r(end) = \emptyset$, $a.end \simeq end$ holds for each $a \in Act$.*

One can intuitively see that the notion of ω-verdict equivalence refers to a form of asymptotic behavior. Indeed, monitors m and n are ω-verdict equivalent if, and only if, they accept and reject the same infinite traces in the sense of [1]. Next we provide a lemma that clarifies the relations between the two notions of equivalence defined above.

Lemma 2. *The following statements hold:*

- $\simeq$ *and* $\simeq_\omega$ *are both congruences.*
- $\simeq \subseteq \simeq_\omega$ *and the inclusion is strict when Act is finite.*
- *If Act is infinite then* $\simeq = \simeq_\omega$.

Proof. For the first claim, it suffices to prove that $\simeq$ and $\simeq_\omega$ are equivalence relations and that they are preserved by $a._-$ and $+$. The proof is standard and is thus omitted.

For the second claim, the inclusion $\simeq \subseteq \simeq_\omega$ is easy to check using the definitions of the two relations. The fact that the inclusion is strict when the set of actions is finite follows from the validity of the equivalence $yes \simeq_\omega \sum_{a \in Act} a.yes$.

However, that equivalence is not valid modulo verdict equivalence since the first monitor accepts the empty string ε, but $\sum_{a \in Act} a.yes$ cannot.

Finally, suppose that Act is infinite. Assume that m and n are ω-verdict equivalent and that s is a finite trace accepted by m. We will argue that n also accepts s. To this end, note that, since Act is infinite, there is some action a that does not occur in m and n. Since m accepts s, the infinite trace sa^ω is in $L_a(m) \cdot Act^\omega$. By the assumption that m and n are ω-verdict equivalent, we have that sa^ω is in $L_a(n) \cdot Act^\omega$. As a does not occur in n, it is not hard to see that n accepts s. Therefore, by symmetry, m and n accept the same traces. The same argument shows that $L_r(m) = L_r(n)$, and therefore $m \simeq n$. □

Equational logic. An axiom system $\mathcal{E}$ over Mon_F is a collection of equations $m = n$ expressed in the syntax of Mon_F. An equation $m = n$ is derivable from an axiom system $\mathcal{E}$ (notation $\mathcal{E} \vdash m = n$) if it can be proven from the axioms in $\mathcal{E}$ using the rules of equational logic (reflexivity, symmetry, transitivity, substitution and closure under the Mon_F contexts). See Table 2. In the rest of this work we shall always implicitly assume, without loss of generality, that equational axiom systems are closed with respect to symmetry, i.e., that if $m = n$ is an axiom, so is $n = m$.

We say that $\mathcal{E}$ is *sound* with respect to $\simeq$ when $m \simeq n$ holds whenever $\mathcal{E} \vdash m = n$. We say that $\mathcal{E}$ is *complete* with respect to $\simeq$ when $\mathcal{E}$ can prove all the valid equations $m \simeq n$. Similar definitions apply for ω-verdict equivalence. The notion of completeness, when limited to closed terms, is referred to as *ground completeness*.

Reflexivity	$t = t$
Symmetry	$\dfrac{t = t'}{t' = t}$
Transitivity	$\dfrac{t_1 = t_2,\ t_2 = t_3}{t_1 = t_3}$
Congruence (For any n-ary f)	$\dfrac{t_i = t_i',\ i = 1, 2, \ldots, n}{f(t_1, \ldots t_n) = f(t_1', \ldots, t_n')}$
Substitutivity (For each substitution σ)	$\dfrac{t = t'}{\sigma(t) = \sigma(t')}$

Table 2: Rules of equational logic

3. A ground-complete axiomatization of verdict and ω-verdict equivalence

Our goal in this paper is to study the equational theory of $\simeq$ and $\simeq_\omega$ over Mon_F. Our first main result is to give a ground-complete axiomatization of verdict equivalence over Mon_F. To this end, consider the axiom system $\mathcal{E}_v$, whose axioms are listed in Table 3.

(A1) $x + y = y + x$	$(\mathbf{E_a})$ $a.end = end$ $(a \in Act)$
(A2) $x + (y + z) = (x + y) + z$	$(\mathbf{Y_a})$ $yes = yes + a.yes$ $(a \in Act)$
(A3) $x + x = x$	$(\mathbf{N_a})$ $no = no + a.no$ $(a \in Act)$
(A4) $x + end = x$	$(\mathbf{D_a})$ $a.(x + y) = a.x + a.y$ $(a \in Act)$

Table 3: The axioms of $\mathcal{E}_v$

Remark 4. *Note that $\mathcal{E}_v$ is finite, if so is Act.*

The subscript $_v$ in the naming scheme of the axiom set refers to the kind of equivalence that it axiomatizes, namely verdict equivalence. It will later be replaced with $_\omega$ when we study ω-verdict equivalence and used accordingly from that point forward.

We provide now the following lemma as an observation on the number of necessary axioms when Act is finite and as an example proof based on these axioms.

Lemma 3. *When Act is finite, the family of axioms (Y_a) can be replaced with*

$$(\mathbf{Y})\ yes = yes + \sum_{a \in Act} a.yes.$$

Similarly the family of axioms (N_a) *can be replaced with*

$$(\mathbf{N})\ no = no + \sum_{a \in Act} a.no.$$

Proof. It is not hard to see that the equation Y can be proved by using the family of equations Y_a. For the converse we can use axioms $A3$ and Y to prove any equation $yes = yes + b.yes$ of the family $\{Y_a \mid a \in Act\}$. Indeed, $\mathcal{E}_v$ proves

$$yes = yes + \sum_{a \in Act} a.yes = yes + \sum_{a \in Act} a.yes + b.yes = yes + b.yes.$$

□

Theorem 1. *$\mathcal{E}_v$ is sound modulo $\simeq$. That is, if $\mathcal{E}_v \vdash m = n$ then $m \simeq n$, for all $m, n \in Mon_F$.*

Proof. It suffices to prove soundness for each of the axioms separately. The details of the proof are standard and therefore omitted. □

In what follows, we will consider terms up to axioms $A1$-$A4$.

A fact that will be proven useful later on is the following: If $m \xrightarrow{a} n$ then $A1 - A4, E_a, Y_a, N_a \ \vdash m = m + a.n$. This follows easily by induction on the size of m and a case analysis on its form and it is thus omitted.

We will now prove that the axiom system $\mathcal{E}_v$ is ground complete for verdict equivalence.

Theorem 2. *$\mathcal{E}_v$ is ground complete for $\simeq$ over Mon_F. That is, if m, n are closed monitors in Mon_F and $m \simeq n$ then $\mathcal{E}_v \vdash m = n$.*

As a first step towards proving that $\mathcal{E}_v$ is complete over closed terms, we isolate a notion of normal form for monitors and prove that each closed monitor in Mon_F can be proved equal to a normal form using the equations in $\mathcal{E}_v$.

Definition 4. ***(Normal Form)*** *A normal form is a closed term $m \in Mon_F$ of the form:*

$$\sum_{a \in A} a.m_a\ [+yes]\ [+no]$$

for some finite $A \subseteq Act$, where each m_a is a term in normal form that is different from end.

Note that, by taking $A = \emptyset$ in the definition above, we obtain that *end* is a normal form. In fact, it is the normal form with the smallest size.

Lemma 4. *The only normal form that does not contain occurrences of yes and no is end.*

Proof. We proceed by induction on the size of a normal form m. Our base case is a verdict v. The only such verdict that does not contain an occurrence of either *yes* or *no* is *end*, which trivially satisfies the lemma. Assume now that $m = \sum_{a \in A} a.m_a$ is a normal form satisfying the statement of the lemma. Since each m_a is *yes*- and *no*-free, by inductive hypothesis, $m_a = end$. This is only possible if $A = \emptyset$. Thus $m = end$. □

Lemma 5. ***(Normalization)*** *Each closed term $m \in Mon_F$ is provably equal to some normal form m' with $depth(m') \leq depth(m)$.*

Proof. We prove the claim by induction on the lexicographic ordering $\prec$ over pairs $(depth(m), size(m))$ of a monitor m, where $size(m)$ denotes the length of m in symbols. We proceed with a case analysis on the form m may have. Our induction basis will be a verdict v. If $v = end$ then the monitor is already in normal form. Otherwise: If $m = v$ for some verdict $v = yes$ or no then it is proved equal to $v + end$ (from axiom $A4$). Indeed this normal form of a non-*end* verdict has depth less or equal to that of the initial monitor.

Our induction hypothesis is that, for all monitors $m_0 \in Mon_F$ up such that $(depth(m_0), size(m_0)) \prec (depth(m), size(m))$, we have that $\mathcal{E}_v \vdash m_0 = m_0'$ with m_0' in normal form and $depth(m_0') \leq depth(m_0)$.

Assume that $m = a.n$ then clearly n has depth less than that of m and therefore by the inductive hypothesis $\mathcal{E} \vdash n \simeq n'$ where n' is in normal form and of depth less or equal than n. If $n' = end$ then $\mathcal{E}_v \vdash m = end$ (using E_a) which is a normal form of smaller depth. Otherwise $a.n'$ is also a normal form.

Assume that $m = m_1 + m_2$. By applying the induction hypothesis we have that

$$\mathcal{E}_v \vdash m_1 = \sum_{a \in A_1} a.m_{1a}\ [+yes][+no] \text{ and } \mathcal{E}_v \vdash m_2 = \sum_{a \in A_2} a.m_{2a}\ [+yes][+no].$$

Therefore by applying axioms from $\mathcal{E}_v$ we can rewrite m as:

$$m = \sum_{a \in A_1 \setminus A_2} a.m_{1a}' + \sum_{a \in A_2 \setminus A_1} a.m_{2a}' + \sum_{b \in A_1 \cap A_2} b.(m_{1b}' + m_{2b}')\ [+yes][+no].$$

Where by the statement of the lemma we have:

$$depth\left(\sum_{a \in A_1 \setminus A_2} a.m_{1a}'\right) \leq depth(m_1) \leq depth(m)$$

and similarly:

$$depth\left(\sum_{a \in A_2 \setminus A_1} a.m_{2a}'\right) \leq depth(m_2) \leq depth(m).$$

It remains to show that the summand $\sum_{b \in A_1 \cap A_2} b.(m'_{1b} + m'_{2b})$ is equal to a normal form and that it has depth less or equal to that of m. However, this is not trivial to see, since the terms m'_{1a} and m'_{2b} have been rewritten by the normalization procedure and therefore we cannot guarantee that their summation has size less of that of m (applying the inductive hypothesis only results in terms of smaller depth but not size as we saw for instance in the case of normalization of verdicts). However we have the following:

$$\begin{aligned} depth(m'_{1b} + m'_{2b}) &= max[depth(m'_{1b}), depth(m'_{2b})] \\ &< max[depth(m'_1), depth(m'_2)]. \end{aligned}$$

The later of the above quantities is guaranteed to be less than or equal to $depth(m)$ by the inductive hypothesis. Therefore we still have that the monitor $m'_{1b} + m'_{2b}$ appears earlier in the lexicographic ordering and therefore $\mathcal{E}_v$ can prove it equal to a normal form of smaller depth. We will call this normal form m'_b. We have therefore that that $depth(m'_b) \leq depth(m'_{1b} + m'_{2b})$. We now have the necessary result that

$$\mathcal{E}_v \vdash m = m' = \sum_{a \in A_1 \cup A_2} a.m_a \ [+yes][+no],$$

where each m_a is in normal form and of depth strictly less than that of m which means that $depth(m') \leq depth(m)$ and we are done. □

Since now we have that each term in Mon_F is provably equal to a normal form, we might attempt to prove Theorem 2 by arguing that the normal forms of two verdict equivalent monitors are identical. However, it turns out that this is not true. Consider, for example, the case were $m = yes$ and $n = yes + a.a.a.yes$. These two monitors are clearly verdict equivalent as $L_a(m) = L_a(n) = Act^*$ and $L_r(m) = L_r(n) = \emptyset$. However, even though they are in normal form they are not syntactically equal. Intuitively, $a.a.a.yes$ in monitor n is redundant, as it can be absorbed by yes. In what follows, we will show how to reduce the normal form of a monitor further using equations in $\mathcal{E}_v$ in order to eliminate such redundant sub-terms.

Lemma 6. *The following statements hold for any monitor in Mon_F:*

1. *For each action a, if m is a closed no-free term then $\mathcal{E}_v \vdash yes + a.m = yes$.*
2. *For each action a, if m is a closed monitor that contains occurrences of both yes and no then $\mathcal{E}_v \vdash yes + a.m = yes + a.n$ for some yes-free closed monitor n.*
3. *For each action a, if m is a closed yes-free term then $\mathcal{E}_v \vdash no + a.m = no$.*
4. *For each action a, if m is a closed monitor that contains occurrences of both yes and no then $\mathcal{E}_v \vdash no + a.m = no + a.n$ for some no-free closed monitor n.*

Proof. We only prove statements *1* and *2* as the proofs of *3* and *4* are similar. We will use structural induction on m.

1. If m is a verdict other than *no* then the claim follows using axioms E_a, Y_a and $A4$ appropriately. If $m = b.m'$ where m' is *no*-free then $\mathcal{E}_v$ derives:

   $$yes+a.m \overset{\text{Y}_\text{a}}{=} yes+a.yes+a.m \overset{\text{D}_\text{a}}{=} yes+a.(yes+b.m') \overset{\textbf{I.H.}}{=} yes+a.yes \overset{\text{Y}_\text{a}}{=} yes.$$

   If m is of the form $m_1 + m_2$ where m_1, m_2 are *no*-free, then it suffices to apply axiom D_a and the induction hypothesis.

2. Assume that m contains occurrences of both *yes* and *no*. We will show that $\mathcal{E}_v \vdash yes + a.m = yes + a.n$ for some *yes*-free monitor n.

   If $m = v$ for some verdict v then the claim follows vacuously.

   If $m = b.m'$ for some m' that contains both *yes* and *no* then there is some *yes*-free n', such that $n = b.n'$, and $\mathcal{E}_v$ derives :

   $$yes + a.m = yes + a.b.m' \overset{\text{Y}_\text{a}}{=} yes + a.yes + a.b.m' \overset{\text{D}_\text{a}}{=} yes + a.(yes + b.m')$$

   and for some *yes*-free n' s.t. $n = b.n'$:

   $$\overset{\text{I.H}}{=} yes + a.(yes + b.n') \overset{\text{D}_\text{a}}{=} yes + a.yes + a.b.n' \overset{\text{Y}_\text{a}}{=} yes + a.n.$$

   Finally if $m = m_1 + m_2$ then $\mathcal{E}_v$ can derive:

   $$yes + a.(m_1 + m_2) \overset{\text{D}_\text{a}}{=} yes + a.m_1 + a.m_2.$$

   We now isolate the following cases based on what verdicts the monitors $m_i,\ i \in \{1, 2\}$ contain. If any $m_i,\ i \in \{1, 2\}$ is both *yes*- and *no*-free it must be equal to *end* as it is in normal form and therefore $\mathcal{E}_v \vdash yes+a.m_i = yes$. If $m_i,\ i \in \{1, 2\}$ contains occurrences of both *yes* and *no*, then the induction hypothesis yields that

   $$\mathcal{E}_v \vdash yes + a.m_i = yes + a.n_i$$

   for some *yes*-free n_i. If $m_i, i \in \{1, 2\}$ is *yes*-free we already have the result that $\mathcal{E}_v \vdash yes + a.m_i = yes + a.n_i$ for some *yes*-free monitor n_i (which in this case coincides with m_i). Finally, if some m_i is *no*-free then, by statement *1* in the lemma,

   $$\mathcal{E}_v \vdash yes + a.m_i = yes.$$

   Combining these observations, we have that:

   $$\mathcal{E}_v \vdash yes + a.m = yes + a.n_1 + a.n_2$$

   where both n_1 and n_2 are *yes*-free and therefore by axiom D_a:

   $$\mathcal{E}_v \vdash yes + a.m = yes + a.n$$

   for some *yes*-free monitor n.

□

The above lemma suggests the notion of a *reduced* normal form.

Definition 5. ***(Reduced normal form)*** *A reduced normal form is a term*

$$m = \sum_{a \in A} a.m_a \ [+yes] \ [+no]$$

in normal form, where if $v \in \{yes, no\}$ is a summand of m then each m_a is v-free and in reduced normal form.

Remark 5. *Note here that if $\sum_{a \in A} a.m_a + yes + no$ is in reduced normal form then $A = \emptyset$.*

Lemma 7. *Each monitor in normal form is provably equal to a monitor in reduced normal form.*

Proof. The claim follows from Lemma 6, using induction on the depth of the normal form. □

We are now ready to complete the proof of Theorem 2.

Proof of Theorem 2. Since each monitor is provably equal to a reduced normal form (Lemma 7), and by the soundness of $\mathcal{E}_v$ (Theorem 1), it suffices to prove the claim for verdict equivalent reduced normal forms m and n. We proceed by induction on the sum of the sizes of m and n, and a case analysis on the possible form m may have.

1. Assume that $m = yes + no \simeq n$. Since $L_a(m) = L_r(m) = Act^*$, it follows that n has both *yes* and *no* as summands. Since n is in reduced normal form it must be the case that $n = yes + no$, and we are done.

2. Assume that $m = \sum_{a \in A} a.m_a + yes \simeq n$, where, for all $a \in A$, m_a is *yes*-free and in reduced normal form and $n = \sum_{b \in B} b.n_b[+yes][+no]$, where each n_b is in reduced normal form and is v-free, if v is a summand of n. Since $\varepsilon \in L_a(m) \setminus L_r(m)$, we have that *yes* is a summand of n and *no* is not. Thus $n = \sum_{b \in B} b.n_b + yes$, and each n_b is *yes*-free. We claim that:

   (C1) $A = B$ and

   (C2) for all $a \in A$, $m_a \simeq n_a$.

   To prove that $A = B$, we assume that $a \in A$. Since m_a is *yes*-free and different from *end*, there is some $s \in Act^*$ such that $a.s \in L_r(m)$. As $m \simeq n$, we have that $a.s \in L_r(n)$. We conclude that $a \in B$ and $s \in L_r(n_a)$. By symmetry, claim **(C1)** follows.

We now show that $m_a \simeq n_a$ for each $a \in A$. Since m_a and n_a are *yes*-free, $L_a(m_a) = L_a(n_a) = \emptyset$. We pick now some arbitrary $s \in L_r(m_a)$ ($L_r(m_a) \neq \emptyset$ because $m_a \neq \mathit{end}$). This means that $a.s \in L_r(m) = L_r(n)$ and therefore $s \in L_r(n_a)$. The claim follows by symmetry. By the induction hypothesis, $\mathcal{E}_v \vdash m_a = n_a$ for each $a \in A = B$. Therefore

$$m = \sum_{a \in A} a.m_a + \mathit{yes} = \sum_{b \in B} b.n_b + \mathit{yes} = n$$

is provable from $\mathcal{E}_v$ and we are done.

3. We are left with the case where $m = \sum_{a \in A} a.m_a + \mathit{no} \simeq n$ and the case $m = \sum_{a \in A} a.m_a$. The proofs for those cases are similar to the one for case *2* and are thus omitted. □

3.1. Axiomatizing ω-verdict equivalence

When Act is infinite, by Lemma 2 and Theorem 2, $\mathcal{E}_v$ gives a ground-complete axiomatization of ω-verdict equivalence as well. However, when Act is finite, $\mathcal{E}_v$ is not powerful enough to prove all the equalities between closed terms that are valid with respect to ω-verdict equivalence. The new axioms needed to achieve a ground complete axiomatization in this setting are:

$$(\mathbf{Y}_\omega) \quad \mathit{yes} = \sum_{a \in \mathit{Act}} a.\mathit{yes} \qquad\qquad (\mathbf{N}_\omega) \quad \mathit{no} = \sum_{a \in \mathit{Act}} a.\mathit{no}.$$

The resulting axiom system is called $\mathcal{E}_\omega$.

Remark 6. *The soundness of the new axioms is trivially shown since*

$$L_a(\mathit{yes}) \cdot \mathit{Act}^\omega = \mathit{Act}^* \cdot \mathit{Act}^\omega = \mathit{Act}^+ \cdot \mathit{Act}^\omega = L_a(\sum_{a \in \mathit{Act}} a.\mathit{yes}) \cdot \mathit{Act}^\omega$$

while $L_r(\mathit{yes}) = L_r(\sum_{a \in \mathit{Act}} a.\mathit{yes}) = \emptyset$ *(and symmetrically for the* N_ω *equation).*

Theorem 3. *$\mathcal{E}_\omega$ is ground complete for $\simeq_\omega$ over closed terms when Act is finite. That is if m, n are closed monitors in Mon_F and $m \simeq_\omega n$ then $\mathcal{E}_\omega \vdash m = n$.*

Proof. By Lemma 7 we may assume that m and n are in reduced normal form. We will prove the claim by induction on the sizes of m and n for two ω-verdict equivalent monitors m, n in reduced normal form.

We will proceed by a case analysis of the form m may have and limit ourselves to presenting the proof for a few selected cases that did not arise in the proof of Theorem 2.

- Assume that $m = \mathit{yes} + \mathit{no} \simeq_\omega \sum_{a \in A} a.n_a = n$. First of all note that $A = \mathit{Act}$. Indeed if $a \in \mathit{Act} \setminus A$ then $a^\omega \in (L_a(m) \cdot \mathit{Act}^\omega) \setminus (L_a(n) \cdot \mathit{Act}^\omega)$ which contradicts our assumption that $m \simeq_\omega n$. Moreover, it is not hard to see

that, for each $a \in Act$, $L_a(n_a) \cdot Act^\omega = L_r(n_a) \cdot Act^\omega = Act^\omega$. This means that, for each $a \in Act$, $n_a \simeq_\omega yes + no$. By induction, for each $a \in Act$, we have that $\mathcal{E}_\omega \vdash n_a = yes + no$. Thus, $\mathcal{E}_\omega \vdash n = \sum_{a \in Act} a.(yes + no)$. From axiom D_a, $\mathcal{E}_\omega \vdash n = \sum_{a \in Act} a.yes + \sum_{a \in Act} a.no$ which from our two new axioms Y_ω, N_ω yields $\mathcal{E}_\omega \vdash n = yes + no = m$, and we are done.

- Assume that $m = yes + no \simeq_\omega \sum_{a \in A} a.n_a + yes$, with each n_a being *yes*-free and different from *end*. Again, reasoning as in the previous case, we have that $A = Act$. Moreover for each $a \in Act$, $L_r(n_a) \cdot Act^\omega = Act^\omega$. Following the same argument as above only for the *no* verdict we arrive at the conclusion that $\mathcal{E}_\omega \vdash n = yes + \sum_{a \in Act} a.no = yes + no = m$.

- The case $m = yes + no \simeq_\omega \sum_{a \in A} a.n_a + no$ is symmetrical to the one above.

- Assume that $m = yes + \sum_{a \in A} a.m_a \simeq_\omega \sum_{b \in B} b.n_b$ where both m and n are in reduced normal form. First of all, we follow an argument similar to the first case analyzed above, to the point where $\mathcal{E}_\omega \vdash n = yes + \sum_{b \in B'} b.n'_b$ for some *yes*-free monitors n'_b. For the proof of this final case we will use the following facts, whose validity can be easily established:

  (**S1**) $B = Act$,

  (**S2**) for all $b \in Act$, $L_a(n_b) = Act^\omega$, and

  (**S3**) for all $a \in A$, $L_r(m_a) = L_r(n_a)$.

  So, for each $a \in A$, $yes + m_a \simeq_\omega n_a$. Since both of these monitors have smaller depth that the original ones, we have that by induction:

  $$\mathcal{E}_\omega \vdash yes + m_a = n_a, \ \forall\ a \in A. \tag{1}$$

  For each $b \in Act \setminus A$, we have that $yes \simeq_\omega n_b$ (because $L^\omega_r(n_b) = \emptyset$). Again, we have that, by induction:

  $$\mathcal{E}_\omega \vdash yes = n_a, \ \forall\ b \in Act \setminus B. \tag{2}$$

  So:

  $$\mathcal{E}_\omega \vdash n = \sum_{b \in Act} b.n_b = \sum_{a \in A} a.n_a + \sum_{b \in Act \setminus A} b.yes$$

  By equations (1) and (2):

  $$\mathcal{E}_\omega \vdash n = \sum_{a \in A} a.(yes + m_a) + \sum_{b \in Act \setminus A} b.yes$$

$$= \sum_{a \in A} a.yes + \sum_{a \in A} a.m_a + \sum_{b \in Act \setminus A} b.yes$$

$$= \sum_{a \in Act} a.yes + \sum_{a \in A} a.m_a = yes + \sum_{a \in A} a.m_a,$$

using axiom Y_ω, and we are done.

The above analysis can be applied symmetrically for the cases:

- $m = no + \sum_{a \in A} a.m_a \simeq_\omega \sum_{b \in B} b.n_b = n$ and
- $m = \sum_{a \in A} a.m_a \simeq_\omega \sum_{b \in B} b.n_b = n$.

This completes the proof. □

4. Open Terms

Thus far, we have only studied the completeness of equational axiom systems for $\simeq$ and $\simeq_\omega$ over closed terms. However, in our grammar we allow for variables and it is natural to wonder whether the ground-complete axiomatizations we have presented in Theorems 2 and 3 are also complete for verdict equivalence and ω-verdict equivalence over open terms. Unfortunately, this turns out to be false. Indeed, the equation

$$(\mathbf{O1}) \quad yes + no = yes + no + x$$

is valid with respect to $\simeq$ (as both sides trivially accept and reject all traces), but cannot be proved using the equations in $\mathcal{E}_\omega$. This is because all the equations in that axiom system have the same variables on their left- and right-hand sides. Our goal in the remainder of this section is to study the equational theory of $\simeq$ and $\simeq_\omega$ over open terms. Subsection 4.1 will present our results when Act is infinite as this case turns out to be more straightforward. We consider the setting of a finite set of actions in Subsection 4.2. In what follows, we use $\mathcal{E}'_v$ for the axiom system that results by adding $O1$ to $\mathcal{E}_v$. The superscript $'$ will be used in the name of an axiom set to denote that the axiom set is complete for one notion of equivalence over *open* terms. The absence of a superscript refers respectively to a *ground complete* axiom set.

Towards a completeness theorem, we modify the notion of normal form, to take variables into account. To that end we define:

Definition 6. *A term $m \in Mon_F$ is in **open normal form** if it has the form:*

$$m = \sum_{a \in A} a.m_a + \sum_{i \in I} x_i \ [+yes] \ [+no]$$

where $\{x_i \mid i \in I\}$ is a finite set of variables, A is a finite subset of Act and each m_a is an (open) term in open normal form that is different from end.

Lemma 8. *Each open term $m \in Mon_F$ is provably equal to some open normal form m' with $depth(m') \leq depth(m)$.*

The proof of the above result follows the lines of the one for Lemma 5 for closed terms and is thus omitted.

As in the case of closed terms, we now proceed to characterize a class of open normal forms for open terms whose verdict equivalence can be detected "structurally". The following example highlights the role that equation $(O1)$ plays in that characterization.

Example 2. *Consider the following monitor in open normal form:*

$$m = x + yes + a.b.(no + b.a.x).$$

Monitor m contains two occurrences of the variable x. However, because of the interplay between the two verdicts, one of them is redundant and can be removed thus:

$$\begin{aligned}
\mathcal{E}'_v \vdash m &= x + yes + a.b.(no + b.a.x) \\
&\overset{\mathrm{Y_a}}{=} x + yes + a.yes + a.b.(no + b.a.x) \\
&\overset{\mathrm{Y_b}}{=} x + yes + a.(yes + b.yes) + a.b.(no + b.a.x) \\
&\overset{\mathrm{D_a}}{=} x + yes + a.(yes + b.yes + b.(no + b.a.x)) \\
&\overset{\mathrm{D_b}}{=} x + yes + a.(yes + b.(yes + no + b.a.x)) \\
&\overset{\mathrm{O_1}}{=} x + yes + a.(yes + b.(yes + no)) \\
&\overset{\mathrm{D_b}}{=} x + yes + a.(yes + b.yes + b.no) \\
&\overset{\mathrm{Y_b}}{=} x + yes + a.(yes + b.no) \\
&\overset{\mathrm{D_a}}{=} x + yes + a.yes + a.b.no \\
&\overset{\mathrm{Y_a}}{=} x + yes + a.b.no.
\end{aligned}$$

The above example motivates the following notion of reduced normal form for open terms.

Definition 7. *An **open reduced normal form** is a term*

$$m = \sum_{a \in A} a.m_a + \sum_{i \in I} x_i \ [+yes] \ [+no]$$

where if $v \in \{yes, no\}$ is a summand of m then each m_a is v-free, different from end and in open reduced normal form. In addition:

- *if both yes and no are summands of m then m is equal to $yes + no$,*

- *if yes is a summand of m and $m \xrightarrow{s} no + m'$, for some s and m' then m' is equal to end,*
- *if no is a summand of m and $m \xrightarrow{s} yes + m'$, for some s and m' then m' is equal to end.*

In what follows we will omit the word "open" when referring to the normal form of a term that contains variables.

Lemma 9. *For each open monitor $m \in Mon_F$, its normal form is provably equal to a reduced normal form.*

Proof. By Lemma 8 we may assume that m is in in normal form. The proof is by induction on the size of m and we isolate the following cases, depending on the verdicts $v \in \{yes, no\}$ m has as summands:

1. Case $m = \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$. In this case we use the induction hypothesis on the m_a monitors. These are different from *end* and have smaller size than m and therefore they are provably equal to a reduced normal form, i.e $\mathcal{E}'_v \vdash m_a = m'_a$ where m'_a is in reduced normal form. Thus $\mathcal{E}'_v$ proves $m = \sum_{a \in A} a.m'_a + \sum_{i \in I} x_i$, and we are done since $\sum_{a \in A} a.m'_a + \sum_{i \in I} x_i$ is in reduced normal form.

   By applying the congruence closure equational law we have that $\mathcal{E}'_v \vdash m = m'$, where m' is in reduced normal form.

2. Case $m = yes + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$. In this case by the induction hypothesis each m_a is provably equal to a reduced normal form. The extra step here is that if $m \xrightarrow{s} no + m'$, for some s and m' then m' is equal to *end*. In such a scenario we have that:

   If $s = \varepsilon$, then the claim follows trivially from $O1$. Otherwise $s = a.s'$ for some action $a \in Act$ and $m \xrightarrow{a} m_a \xrightarrow{s'} no + m'$. We now apply our axioms as follows:

$$m = yes + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i \overset{\mathrm{Y_a}}{=} yes + \sum_{b \in A \setminus \{a\}} b.m_b + a.yes + a.m_a + \sum_{i \in I} x_i$$

$$\overset{\mathrm{D_a}}{=} yes + \sum_{b \in A \setminus \{a\}} b.m_b + a.(yes + a.m_a) + \sum_{i \in I} x_i.$$

   This means that since $yes + m_a$ has size smaller than m it is provably equal to a reduced normal form. Additionally, since it contains a *yes* summand and $m_a \xrightarrow{s'} no + m'$, by the induction hypothesis we have that m' is equal to *end* and we are done.

3. Case $m = no + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$. The proof of this case is symmetrical to Case 2 and therefore omitted.

4. Case $m = yes + no + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$. In this case we use the following simple argument. Starting for axiom O_1 we use the substitution $\sigma(x) = \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$ and we get:

$$yes + no = yes + no + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i,$$

and by applying the equational law of transitivity we have that $\mathcal{E}'_v \vdash m = yes + no$. □

The normal form defined above for open terms is adjusted over the closed terms case. This is because now our syntax is allowing for variables and therefore it is convenient for proofs to take these variables into account in a controlled and consistent manner. The further reducing that occurred towards defining the open reduced normal forms was possible due to the existence of the new axiom O_1, which gave us the option to remove variable occurrences. The new axiom O_1 is the only axiom we have currently available that does not contain every variable occurrence in both of its sides and it is therefore the only rule we have available that can help us remove variables from equations. In the presence of other axioms with this property we can further reduce our normal forms, as we will see later on.

In the following subsections, we will study the full equational theory of verdict and omega-verdict equivalence over open terms.

4.1. Infinite set of actions

We begin by considering the equational theory of open monitors when the set of actions is infinite. Apart from its theoretical interest, this scenario has also some practical relevance. Indeed, as shown already by Milner in [42, 44], infinite sets of uninterpreted actions are useful when modeling system events that carry data values. Runtime monitoring of systems with data-dependent behavior has been an active field of research for over 15 years—see, for instance, the paper [11] for an early reference.

When the set of actions Act is infinite, it is easy to define a one-to-one mapping from open to closed terms that will help us prove completeness of the axiom system $\mathcal{E}'_v$.

Theorem 4. *(Completeness for open terms modulo $\simeq$) $\mathcal{E}'_v$ is complete for $\simeq$ over open monitors in Mon_F when Act is infinite. That is, for all $m, n \in Mon_F$, if $m \simeq n$, then $\mathcal{E}'_v \vdash m = n$.*

Proof. Assume $m \simeq n$. By Lemma 9, we may assume that m and n are in reduced normal form.

Let

$$m = \sum_{a \in A} a.m_a + \sum_{i \in I} x_i \ [+yes] \ [+no]$$

and

$$n = \sum_{b \in B} b.n_b + \sum_{j \in J} y_j \ [+yes] \ [+no].$$

We will show that $\mathcal{E}'_v \vdash m = n$ by induction on the sum of the sizes of m and n. To this end, we will establish a strong structural correspondence between m and n. Consider a substitution σ defined as follows: $\sigma(x) = a_x.(yes + no)$ where

- for all variables x and y, $a_x = a_y$ implies $x = y$, and
- $\{a_x \mid x \in Var\}$ is disjoint from the set of actions occurring in m or n.

Note that such a substitution σ exists because Act is infinite. By induction on the sizes of m and n, we will prove that if $\sigma(m) \simeq \sigma(n)$ then:

(**C1**) v is a summand of m iff v is a summand of n, for $v \in \{yes, no\}$,

(**C2**) $\{x_i \mid i \in I\} = \{y_j \mid j \in J\}$,

(**C3**) $A = B$ and

(**C4**) for each $a \in A$, $\sigma(m_a) \simeq \sigma(n_a)$.

In what follows, we first show that $\mathcal{E}'_v$ proves $m = n$ assuming claims **(C1)-(C4)** and then we prove those claims. To prove that $\mathcal{E}'_v$ proves $m = n$ follows from $\sigma(m) \simeq \sigma(n)$ for reduced normal forms m and n, we proceed by induction on the sum of the sizes of m and n. By claim **C4**, we have that $\sigma(m_a) \simeq \sigma(n_a)$ and, from the induction hypothesis, $\mathcal{E}'_v \vdash m_a = n_a$. By **C1-3** we also have that $\mathcal{E}'_v \vdash \sum_{i \in I} x_i = \sum_{j \in J} y_j$ and that $\sum_{a \in A} a.m_a = \sum_{b \in B} b.n_b$, which means that by using the equational law of closure under summation we also have that $\mathcal{E}'_v \vdash m = n$.

We present now the proofs of **(C1)-(C4)**.

C1: Assume yes is a summand of m. Then $\varepsilon \in L_a(\sigma(m))$. Since $\sigma(m) \simeq \sigma(n)$, we have that $\varepsilon \in L_a(\sigma(n))$. Note that $\varepsilon \notin L_a(\sigma(x))$ for each x. Thus yes must be a summand of n. The case for $v = no$ is similar. By symmetry the claim follows.

C2: Assume that $x \in \{x_i \mid i \in I\}$. By the definition of σ, it follows that $\sigma(m)$ both accepts and rejects the trace a_x. Since $m \simeq n$, we have that $\sigma(n)$ also accepts and rejects the trace a_x. As n does not contain any occurrence of a_x and has at most one of the verdicts yes and no as a summand, it follows that $x \in \{y_j \mid j \in J\}$. Therefore, by symmetry, $\{x_i \mid i \in I\} = \{y_j \mid j \in J\}$ and we are done.

C3: Assume, towards a contradiction, that $a \in A \setminus B$. Then m cannot have both yes and no as summands, since m is in reduced normal form.

If m has none of the verdicts as a summand, we know that m_a is different from *end* since m is in reduced normal form. Therefore $\sigma(m_a)$ will either accept or reject some trace s, which implies that $\sigma(m)$ will also accept or reject as. However, $\sigma(n)$ cannot do the same because $a \notin B$, $\sigma(x) \stackrel{a}{\not\Rightarrow}$ for each x, and neither *yes* nor *no* are summands of n. This contradicts our assumption that $m \simeq n$.

Assume now, without loss of generality, that m has only the verdict *yes* as summand. Observe that m_a is *yes*-free and different from *end*, since m is in reduced normal form. This means that $\sigma(m_a)$ can reject some trace s and, therefore, that $\sigma(m)$ will reject $a.s$. On the other hand, $\sigma(n)$ cannot do the same because $a \notin B$, $\sigma(x) \stackrel{a}{\not\Rightarrow}$ for each x and *no* is not a summand of n. Again, this contradicts our assumption that $m \simeq n$.

The above analysis yields that $A \subseteq B$. By symmetry, $A = B$ follows.

C4: Our final claim (and the one with the most involved proof) is that $\sigma(m_a) \simeq \sigma(n_a)$, for each $a \in A$.

If the reduced normal forms of the monitors do not contain any verdict $v \in \{yes, no\}$ as a summand, then the argument is simplified significantly. Therefore, we limit ourselves to presenting here the most complicated case, where m and n both contain exactly one verdict $v \in \{yes, no\}$ as a summand. Without loss of generality, we assume that this verdict is *yes*, namely that

$$m = yes + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$$

and

$$n = yes + \sum_{b \in B} b.n_b + \sum_{j \in J} y_i.$$

Since the claims **C1-3** have already been proven, we know for m and n that:

$$m = yes + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i \text{ and } n = yes + \sum_{a \in A} a.n_a + \sum_{i \in I} x_i \ .$$

We remind the reader that our purpose is to prove that $\sigma(m_a) \simeq \sigma(n_a)$, for each $a \in A$, so that we can apply our induction hypothesis to infer that $\mathcal{E}'_v \vdash m_a = n_a$.

We first prove that the rejection sets of $\sigma(m_a)$ and $\sigma(n_a)$ are equal. To this end, assume that $s \in L_r(\sigma(m_a))$. It follows that $a.s \in L_r(\sigma(m)) = L_r(\sigma(n))$. By the form of n and from the definition of σ, we conclude that $s \in L_r(\sigma(n_a))$. Therefore, $L_r(\sigma(m_a)) \subseteq L_r(\sigma(n_a))$. By symmetry we have that $L_r(\sigma(m_a)) = L_r(\sigma(n_a))$ and we are done.

It remains to prove that the acceptance sets of $\sigma(m_a)$ and $\sigma(n_a)$ are also identical. (It is important here to point out that, since both m and n contain a *yes* verdict as a summand, the acceptance sets of $\sigma(m)$ and $\sigma(n)$ are both equal to Act^*. However, for our inductive argument to work, we need to be able to prove that $L_a(\sigma(m_a)) = L_a(\sigma(n_a))$.) To that end and towards a contradiction, consider a shortest trace s that is accepted by monitor $\sigma(m_a)$, but not by $\sigma(n_a)$. Consequently, monitor $\sigma(m)$ accepts the trace $a.s$.

Since monitors m_a and n_a are *yes*-free, as a result of m and n being in reduced normal form, the acceptance of s must be the result of a variable x mapped to $a_x.(yes + no)$ through the substitution σ. Since s is a shortest trace that is accepted by monitor $\sigma(m_a)$, but not by $\sigma(n_a)$, none of its prefixes is accepted by $\sigma(m_a)$ and therefore the last action that is in s must be the action a_x stemming from $\sigma(x)$. This means that monitor m_a can perform the transition $m_a \stackrel{s'}{\Rightarrow} m'_a$, where m'_a contains x as a summand and $s = s'.a_x$. Therefore the monitor $\sigma(m_a)$ can perform the transitions:

$$\sigma(m_a) \stackrel{s'}{\Rightarrow} \sigma(m'_a) \xrightarrow{a_x} yes + no \xrightarrow{\tau} yes.$$

Since $s'.a_x$ is accepted by $\sigma(m_a)$, it must also be rejected by it because a_x is an action that can only be observed after the substitution of the variable x in m_a. We have already argued that the rejection sets of $\sigma(m_a)$ and $\sigma(n_a)$ are equal and therefore $\sigma(n_a)$ also rejects the trace $s'.a_x$. Since the action a_x is a unique action corresponding to the variable x, there are only two ways in which $\sigma(n_a)$ could reject the trace $s'.a_x$. The first case is that $\sigma(n_a)$ can also perform the transitions

$$\sigma(n_a) \stackrel{s'}{\Rightarrow} \sigma(n'_a) \xrightarrow{a_x} yes + no$$

wfor some n'_a. However, this would guarantee that $\sigma(n_a)$ accepts s, whereas we assumed that it does not.

The most complicated case is when $\sigma(n_a)$ can reject a prefix s_0 of s'. By the already proven equality of the rejection sets of the two sub-monitors, $\sigma(m_a)$ would also reject s_0. This can only happen if both n_a and m_a rejected that prefix independently of the substitution σ, since every action preceding a_x along the trace s' is not an action corresponding to the mapping of a variable through σ as explained above. This means that both m_a and n_a can perform the transitions $m_a \stackrel{s_0}{\Rightarrow} no + m'_a$ and $n_a \stackrel{s_0}{\Rightarrow} no + n'_a$, for some m'_a and n'_a. However, since m and n are in reduced normal form, this implies that m'_a and n'_a are equal to *end*. This leads us to a contradiction, as we assumed that $\sigma(m_a)$ accepts the trace s which can no longer be the case if $m_a \xrightarrow{s_0} no + end$ where s_0 is a prefix of s.

Therefore every trace accepted by $\sigma(m_a)$ is also accepted by $\sigma(n_a)$. By symmetry, we have that the acceptance sets of m_a and n_a are equal.

This means that $\sigma(m_a) \simeq \sigma(n_a)$, which completes the proof of **C4** and consequently of the whole theorem. □

Corollary 1. *$\mathcal{E}'_v$ is complete for $\simeq_\omega$ over open monitors in Mon_F when Act is infinite. That is, for all $m, n \in Mon_F$, if $m \simeq n$, then $\mathcal{E}'_v \vdash m = n$.*

Proof. The claim follows from Lemma 2. □

4.2. Finite set of actions

The study of the equational theory of $\simeq$ when *Act* is finite turns out to be more interesting and complicated. In this setting, we can identify equations

whose validity depends on the cardinality of Act, which is not the case for any of the axioms we used so far. To see this, consider the equation

$$(\mathbf{V_1}) \quad x = x + a.x,$$

which is sound when $Act = \{a\}$ but cannot be derived by the equations in $\mathcal{E}'_v$, as it is not sound when $Act \neq \{a\}$.

As a first step in our study of the equational theory of $\simeq$ when Act is finite, we characterize some properties of sound equations.

Lemma 10. *Let $m \simeq n$ be a sound equation, where $m, n \in Mon_F$ and m is in reduced normal form. Assume that*

- *$m \xrightarrow{s} x + m'$, for some s in Act^*, variable x and m' in Mon_F, and*
- *$m \xrightarrow{s_p} \not\rightarrow x + m_{s_p}$, for each proper prefix s_p of s and $m_{s_p} \in Mon_F$.*

Then, $n \xrightarrow{s} x + n'$ for some n' in Mon_F.

Proof. Consider the substitution

$$\sigma(y) = \begin{cases} yes + no, & \text{if } y = x \\ end, & \text{if } y \neq x. \end{cases}$$

Since $m \xrightarrow{s} x + m'$ by one of the assumptions of the lemma, we have that $\sigma(m)$ will both accept and reject s. Since $m \simeq n$ is sound we have that $\sigma(n)$ must do the same. If $n \xrightarrow{s} \not\rightarrow x + n'$ for every n' then it is not hard to see that there are two ways in which n could accept and reject s:

1. $n \stackrel{s'}{\Rightarrow} yes$ and $n \stackrel{s'}{\Rightarrow} no$ where s' is a prefix of s (including s itself), or
2. $n \xrightarrow{s'} x + n'$ where s' is a prefix of s (so that $\sigma(n)$ would accept and reject s' and therefore s).

In the first case, consider the substitution σ_e that maps all variables to end. Since $n \stackrel{s'}{\Rightarrow} yes$ and $n \stackrel{s'}{\Rightarrow} no$, we have that $\sigma_e(n)$ accepts and rejects s'. From $m \simeq n$, we have that $\sigma_e(m)$ also accepts and rejects s'. It is not hard to see that this means that $m \stackrel{s'}{\Rightarrow} yes$ and $m \stackrel{s'}{\Rightarrow} no$ However, this is impossible because m is a reduced normal form and $m \xrightarrow{s} x + m'$ by the proviso of the lemma.

In the second case, even though both monitors accept and reject s, we also have that $\sigma_e(n)$ also accepts and rejects s'. Again, since the two monitors are verdict equivalent, we know that $\sigma_e(m)$ must do the same. Since m is in reduced normal form and $m \stackrel{s_p}{\not\Rightarrow} x + m'$ for any prefix s_p of s (and therefore neither for s') we have that $\sigma(m)$ can only accept and reject s' by performing the transitions

$m \stackrel{s_1''}{\Longrightarrow} yes$ and $m \stackrel{s_2''}{\Longrightarrow} no$, for s_1'' and s_2'' prefixes of s'. This however is not allowed since it contradicts the fact that m is in reduced normal form and $m \stackrel{s}{\rightarrow} x + m'$. Since both cases have led to a contradiction, we can infer that there is some n' such that $n \stackrel{s}{\Rightarrow} x + n'$, which was to be shown. □

Corollary 2. *Let $m \simeq n$ be a sound equation, where $m, n \in Mon_F$ and m is in reduced normal form. Assume that*

- *$m \stackrel{s}{\rightarrow} x + m'$, for some s in Act^*, variable x and m' in Mon_F, and*
- *$n \stackrel{s}{\not\rightarrow} x + n'$, for any $n' \in Mon_F$.*

then we have that there exists an s_p prefix of s such that

- *$m \stackrel{s_p}{\longrightarrow} x + m_{s_p}$ and $n \stackrel{s_p}{\longrightarrow} x + n_{s_p}$ for some m_{s_p} and n_{s_p} in Mon_F, and*
- *for any prefix s_0 of s_p we have that $m \stackrel{s_0}{\not\Rightarrow} x + m'$ and $n \stackrel{s_0}{\not\Rightarrow} x + n'$ for any m' and n' in Mon_F.*

Proof. Assume a sound equation $m \simeq n$ for witch we have $m \stackrel{s}{\rightarrow} x + m'$, for some s in Act^*, variable x and m' in Mon_F. If this is the first occurrence of x along the trace s in m (i.e. $m \stackrel{s_p}{\not\longrightarrow} x + m_{s_p}$, for each proper prefix s_p of s and $m_{s_p} \in Mon_F$), then by Lemma 10, we would have that n must be able to perform the transitions $n \stackrel{s}{\rightarrow} x + n'$, for some n' in Mon_F. Since this cannot be the case as the proviso of the corollary forbids it we have that there must be a prefix s_p of s such that $m \stackrel{s_p}{\longrightarrow} x + m_{s_p}$.

Without loss of generality we assume s_p to be the shortest such trace, which means there are no other occurrences of the variable x along the trace s_p. We can therefore see that now for the trace s_p, Lemma 10 holds and therefore $n \stackrel{s_p}{\longrightarrow} x + n_{s_p}$ for some n_{s_p} in Mon_F. Additionally since we assumed s_p t be the shortest trace of the necessary property we already have that $m \stackrel{s_0}{\not\Rightarrow} x + m'$ for any m' in Mon_F.

It remains to show that the same must hold for n. This can be easily seen to be the case since if we assumed the opposite where for some prefix s_0 of s_p we had $n \stackrel{s_0}{\longrightarrow} x + n_0$ for some n_0 then by the symmetric analysis and by using the previous lemma and this corollary we would arrive at a contradiction of s_p being the shortest prefix of s for witch $m \stackrel{s_p}{\longrightarrow} x + m_{s_p}$. □

Remark 7. *In what follows, when studying open equations, we will refer to occurrences of variables such as the one mentioned in the above corollary, where only one of the monitors involved in the equation can reach a term of the form $x + m_x$ after observing a trace s, as "one-sided" variable occurrences.*

Intuitively Lemma 10 states that on each sound equation (including axioms) of which at least one side is in reduced normal form, the first occurrence of each variable per distinct trace leading to the variable is common for both sides of the equation.This gives us some handy intuition on what restrictions an equation that is sound must satisfy.

The following example shows Lemma 10 in action.

Example 3. *The equation*

$$x + a.(x + a.(yes + no) + b.(yes + no)) = x + a.(a.(yes + no) + b.(yes + no))$$

is sound over the set of actions $Act = \{a, b\}$*, but*

$$x + a.(x + a.(yes + no) + b.(yes + no)) = a.(x + a.(yes + no) + b.(yes + no))$$

is not since the first occurrence of the variable x *in the second example happens after the prefix* ε *on the left-hand side but after the prefix* a *on the right. In the second equation, the earliest occurrence of the variable* x *(after the prefix* ε*) is one-sided.*

Also notice here the importance of the sub-term $a.\sum_{a\in Act} a.(yes+no)$*. We will see that this type of sub-term is crucial for the soundness of the open equations with one-sided variable occurrences we encounter later on.*

The following notation will be used in what follows to describe a family of sound equations that generalize the one given in Example 3.

Definition 8. *(**Notation**) Let* $s \in Act^*$*.*

1. *We use* $pre(s)$ *to denote the set of prefixes of* s *(including* s*).*
2. *We use* s^i*,* $i \geq 1$*, to denote the trace* s *if* $i = 1$ *and* ss^{i-1} *otherwise.*
3. *We use* $s.m$ *to stand for a monitor that can perform exactly the actions along the finite trace* s *and then become* m*.*
4. *We define* $\overline{s}^{\leq}(m) = \sum_{\substack{|s'|\leq|s|,\\ s'\notin pre(s)}} s'.m$ *. The monitor* $\overline{s}^{\leq}(m)$ *is one that behaves like* m *after having observed any trace of length at most* $|s|$ *that is not a prefix of* s*.*
5. *The term* $\overline{s}(m)$ *is defined thus:* $\overline{s}^{\leq}(m) + s.\sum_{a\in Act} a.m$ *.*

   Intuitively $\overline{s}(yes + no)$ *stands for the monitor that accepts and rejects all traces that* do not *cause the acceptance or rejection of the string* s*. Those are exactly the traces that are shorter than* s *but not its prefixes, and also the ones extending* s*.*

6. *With the term* $\overline{s}^{(k)}(m)$*, for* $k \geq 1$*, we will mean the summation:*

   $\overline{s}(m)$ *if* $k = 1$ *and* $\sum_{1 \leq i < k-1} s^i.\overline{s}^{\leq}(m) + s^{k-1}.\overline{s}(m)$ *if* $k \geq 2$*.*

   Intuitively $\overline{s}^{(k)}(yes + no)$ *stands for a monitor that, after observing the fixed trace* s*, accepts and rejects everything except the trace* s^k *(and its prefixes).*

We now present an example of the usage of the above notation in order to help the reader understand the equations presented later involving these new notions.

Example 4. *For a set of actions* $Act = \{a, b\}$*, the monitor* $m = yes + no$ *and a trace* $s = ab$ *we have that:*

- $pre(s) = \{\varepsilon, a, ab\}$
- $\overline{s}^{\leq}(m) = b.(yes + no) + a.a.(yes + no) + b.b.(yes + no) + b.a.(yes + no)$
- $\overline{s}(m) = b.(yes + no) + a.a.(yes + no) + b.b.(yes + no) + b.a.(yes + no) + a.b.\sum_{c \in Act} c.(yes + no)$
- *and for* $k = 3$ *we get*

$$\overline{s}^{(3)}(m) = s.\overline{s}^{\leq}(m) + s^2.\overline{s}(m) =$$

$$a.b.(b.(yes + no) + a.a.(yes + no)) +$$
$$a.b.a.b.(b.(yes + no) + a.a.(yes + no) + b.b.(yes + no) +$$
$$b.a.(yes + no) + a.b.\sum_{c \in Act} c.(yes + no))$$

This notation defined and presented above is very useful once one understands a very particular form equations among open monitors take when they involve one-sided variable occurrences. Consider, for instance, the following sound equation (for a fixed constant k):

$$x + a^k.x + \overline{a^k}^3(yes + no) \simeq x + \overline{a^k}^3(yes + no)\ .$$

We will formally prove the soundness of (a more general form of) this equation later on. We can intuitively see from the examples above that when an equation contains a one-sided variable occurrence, then the rest of the terms involved in the equation must have some specific form as well so that the equation will stay sound under all possible substitutions. This means that certain traces must always be accepted and rejected by both sides independently of a substitution.

The following lemma formalizes this intuition.

Lemma 11. *Assume $m \simeq n$, where m, n are in reduced normal form. If $m \xrightarrow{s} x + m'$ for some m' but $n \not\xrightarrow{s} x + n'$ for any n', then there exist s', s'' such that $s = s's''$ and, for all $s_b = ss_p$ where $s_p \notin pre(s'')$, either:*

- *$m \stackrel{s_b}{\Rightarrow} yes$, $m \stackrel{s_b}{\Rightarrow} no$, $n \stackrel{s_b}{\Rightarrow} yes$ and $n \stackrel{s_b}{\Rightarrow} no$ or*
- *$\exists s_0, m'', n''$ such that $m \xrightarrow{s_0} x + m''$ and $n \xrightarrow{s_0} x + n''$ and $s_0.s_b \in pre(s.s_b)$.*

Proof. We have an equation $m \simeq n$, with m and n in reduced normal form, for which we assume that: $m \xrightarrow{s} x + m'$ but $n \not\xrightarrow{s} x + n'$ for any n'. Let s be the shortest trace meeting the proviso of the lemma. It is not hard to see that $s \neq \varepsilon$ because $m \simeq n$ and m and n are in reduced normal form. This means that indeed in the monitors m, n all other earlier occurrences of x happen at both sides. By Corollary 2 we know that there is a prefix of s called s' ($s = s'.s''$) such that both m and n can perform the transitions $m \xrightarrow{s'} x + m'_0$ and $n \xrightarrow{s'} x + n'_0$, and in addition for every prefix of s' we have that $n \not\xrightarrow{s'} x + n'$ and $m \not\xrightarrow{s'} x + m'$ for every m' and n'.

This means that there are no other one-sided occurrences of the variable x "between" s' and s'' (otherwise s would not be the shortest trace). Since $m \simeq n$ is sound, we know that under any substitution the resulting monitors are verdict equivalent.

Consider the set of traces

$$A = \{t \mid (|t| \leq |s''| \wedge t \notin pre(s'')) \vee t = s''.t',\ t' \in Act^+\}\ .$$

We now associate with this set of traces the class $\mathcal{S}_A$ of substitutions σ as the ones that for at least one trace $s_p \in A$ we have that $\sigma(x) \stackrel{s_p}{\Rightarrow} yes$ or $\sigma(x) \stackrel{s_p}{\Rightarrow} no$. Note that the class of substitution $\mathcal{S}_A$ contains many substitution for each trace s_p and, additionally, since the set A is infinite, $\mathcal{S}_A$ is infinite as well.

Fix now a s_p and a substitution $\sigma \in \mathcal{S}_A$ such that $\sigma(x) \stackrel{s_p}{\Rightarrow} yes$. We have therefore that $\sigma(m) \stackrel{s's_p}{\Longrightarrow} yes$, $\sigma(n) \stackrel{s's_p}{\Longrightarrow} yes$ and $\sigma(m) \stackrel{ss_p}{\Longrightarrow} yes$. By the construction of A, $s's_p$ is not a prefix of ss_p and therefore it is not necessary that $\sigma(n) \stackrel{ss_p}{\Longrightarrow} yes$. However, since $m \simeq n$ is sound we have that $\sigma(n)$ must also be able to accept $s.s_p$. One way this could happen is if both monitors, m and n accept and reject the trace $s_b = s.s_p$ where $s_p \in A$ independently of a substitution, i.e. $m \stackrel{s_b}{\Rightarrow} yes$, $m \stackrel{s_b}{\Rightarrow} no$, $n \stackrel{s_b}{\Rightarrow} yes$ and $n \stackrel{s_b}{\Rightarrow} no$. Note that if one monitor can perform these transitions independently of a substitution then the other one must do so as well since they are verdict equivalent. If this is the case then for the traces s', s'' with $s = s's''$ an for all $s_b = ss_p$ where $s_p \notin pre(s'')$ the first bullet of the lemma holds.

If this is not the case however we have that for a trace s_p and a substitution $\sigma \in \mathcal{S}_A$ such that $\sigma(x) \stackrel{s_p}{\Rightarrow} yes$ the monitor n must somehow accept the trace ss_p and this is not done because $n \stackrel{s_b}{\Rightarrow} yes$.

We remind to the reader here that s is the shortest we could find that satisfied the proviso of the lemma. Therefore there are no other one-sided variable occurrences along the trace s.

This means that the only way than n could accept s_b is another variable occurrence (not one-sided as s is the shortest trace satisfying the proviso of the the lemma) happening after some other prefix s_0 of s. I.e. $n \xrightarrow{s_0} x + n_1$, $m \xrightarrow{s_0} x + m_1$ for some monitors n_1 and m_1 and trace $s_0.s_p$ is a prefix of $s'.s''.s_p = s.s_p$. Note here that by Corollary 2 we know that s' is the shortest trace after which the variable x occurs. Therefore our only options for the trace s_0 would be the trace s' and its extensions which falls in the second case of the lemma as $s_0.s_p$ is a prefix of $s.s_p$.

This concludes the case analysis for the shortest s leading to a one-sided variable occurrence of a variable. We continue with a trace s_1 as the immediately longer than s. For this s_1 with $|s_1| \geq |s|$ we can generalize the result as follows:

If $s \in pre(s_1)$ then the trace s' we identified with the case analysis s is also a prefix of s_1 (i.e. $s_1 = s'.s_1''$) and the same transitions we proved for the traces s_b are also enough for the result to hold for the trace s_1. Assume now that $s \notin pre(s_1)$. Then Corollary 2 still holds and the one-sided variable occurrence after the trace s_1 also does not have any other one-sided variable occurrences between itself and the prefix guaranteed by the corollary which means we can apply the same analysis. □

4.2.1. Completeness of verdict equivalence

In this section we will present our axiom system for open monitors over a finite number of actions. We start by providing an axiom set, which we prove to be sound and complete for verdict equivalence over Mon_F. In order to do so, we first use these axioms to further reduce a normal form of a term. Then, by utilizing this new reduced normal form we use structural induction to prove the completeness of our axiom set. The axiom set we provide is infinite. It is therefore natural to ask whether $\simeq$ is finitely axiomatizable over Mon_F. We answer this question negatively by proving that no complete finite axiom set exists for this algebra. This final part follows a different type of argument which we will present in Section 5.

When studying open equations over a finite set of actions one would hope that one of the axiom systems presented already would be complete. However, we can guarantee that the equations provided in $\mathcal{E}'_v$ are definitely unable to prove every sound open equation. To see this consider the equation used in Example 4 (where k is a constant):

$$x + a^k.x + \overline{a^k}^{(3)}(yes + no) \simeq x + \overline{a^k}^{(3)}(yes + no) \ .$$

We can clearly see that one of the sides of this equations contains a one-sided variable occurrence (remember that we are considering terms up to $A1 - A4$). The only axiom which has a similar behavior is $O1$. However for axiom $O1$ to be applied it must be the case that a variable is occurring simultaneously with a *yes and* a *no* verdict. Since this does not apply for the equation we are examining it is easy to see that no proof involving only the axioms of $\mathcal{E}'_v$ could prove it.

Towards proving this kind of equations and when Act is finite, we consider the family of axioms

$$\mathcal{O} = \{O2_{s,k} \mid s \in Act^*, k \geq 0\}$$

where

$$(\mathbf{O2_{s,k}}) \quad x + s.x + \overline{s}^{(k)}(yes + no) = x + \overline{s}^{(k)}(yes + no) \ .$$

We extend our finite axiom set $\mathcal{E}'_v$ for open terms to the infinite $\mathcal{E}'_v \cup \mathcal{O}$, which we will call $\mathcal{E}'_{v,f}$. The subscript f in the naming scheme states that the action set for which the axiom system is complete is finite. When the action set is a singleton, we will replace it with the subscript 1. If the cardinality of the action set is not important, or if it is infinite, then we use no subscript. Based on the naming scheme we have defined, the name of the axiom set $\mathcal{E}'_{v,f}$ denotes that we are studying verdict equivalence $(_v)$, over open terms $(')$ and for a finite set of actions $(_f)$.

Lemma 12. *$\mathcal{E}'_{v,f}$ is sound. That is, if $\mathcal{E}'_{v,f} \vdash m = n$ then $m \simeq n$, for all $m, n \in Mon_F$.*

Proof. We have to prove soundness only for the new family of equations $\mathcal{O}$ as the other equations are sound by Theorem 1.

First of all, note that $\sigma(x+s.x+\overline{s}^{(k)}(yes+no))$ accepts every trace accepted by $\sigma(x+\overline{s}^{(k)}(yes+no))$, and rejects every trace rejected by $\sigma(x+\overline{s}^{(k)}(yes+no))$. We are therefore left to show that

- if $\sigma(s.x)$ accepts some trace then so does $\sigma(x + \overline{s}^{(k)}(yes + no))$, and
- if $\sigma(s.x)$ rejects some trace then so does $\sigma(x + \overline{s}^{(k)}(yes + no))$.

We only detail the proof for the latter claim, as that of the former one is similar. To this end, assume that $\sigma(s.x)$ rejects some trace s'. Then $s' = ss''$ for some s'' that is rejected by $\sigma(s.x)$. If s'' is a prefix of s^k, then it is not hard to see that $\sigma(x)$ rejects s' too, and thus so does $\sigma(x + \overline{s}^{(k)}(yes + no))$. On the other hand, if s'' is not a prefix of s^k, then $s' = ss''$ is not a prefix of s^k either. Therefore, $\sigma(\overline{s}^{(k)}(yes + no))$ rejects s'. It follows that $\sigma(x + \overline{s}^{(k)}(yes + no))$ rejects s', and we are done. □

We provide here some examples of how to use the above to derive some simpler and more intuitive sound equations.

Lemma 13. *The following equations are derivable from $\mathcal{O}$ for each $s, s_1 \in Act^*$:*

1. $x + s.x + s.\overline{s_0}(yes + no) = x + s.\overline{s_0}(yes + no)$, *with* s_0 *a prefix of* s,
2. $yes + x + s_1.\overline{s_2}(no) = yes + x + s_1.\overline{s_2}(no) + s_1.x$, *where* s_2 *is any prefix of* s_1,
3. $no + x + s_1.\overline{s_2}(yes) = no + x + s_1.\overline{s_2}(yes) + s_1.x$, *where* s_2 *is any prefix of* s_1,
4. $x + s.\sum_{a \in Act} a.(no + yes) = x + s.(x + \sum_{a \in Act} a.(no + yes))$.

Proof. We first show how to derive the first equation and then we derive the rest from it. We start by picking the equation $O2_{s,1}$ i.e.

$$x + s.x + s.\overline{s}^{\leq}(yes + no) + s.s.\sum_{a \in Act} a.(yes + no) =$$

$$x + s.\overline{s}^{\leq}(yes + no) + s.s.\sum_{a \in Act} a.(yes + no)\ .$$

In addition we have the tautology

$$s.s_0.\sum_{a \in Act} a.(yes + no) = s.s_0.\sum_{a \in Act} a.(yes + no)\ ,$$

for the specific prefix s_0 of s. On the two valid above equations we apply the congruence rule for $+$ and have:

$$x + s.x + s.\overline{s}^{\leq}(yes + no) + s.s.\sum_{a \in Act} a.(yes + no) + s.s_0.\sum_{a \in Act} a.(yes + no)$$

$$= x + s.\overline{s}^{\leq}(yes + no) + s.s.\sum_{a \in Act} a.(yes + no) + s.s_0.\sum_{a \in Act} a.(yes + no)\ .$$

The first simplification that we perform now is by observing that the summand $s.s_0.\sum_{a \in Act} a.(yes + no)$ accepts and rejects a prefix of the whole summand $s.s.\sum_{a \in Act} a.(yes + no)$ and therefore we can eliminate the latter from the summation:

$$x + s.x + s.\overline{s}^{\leq}(yes + no) + s.s_0.\sum_{a \in Act} a.(yes + no)$$

$$= x + s.\overline{s}^{\leq}(yes + no) + s.s_0.\sum_{a \in Act} a.(yes + no)\ .$$

In addition the term $s.\overline{s}^{\leq}$ can be rewritten as $s.\overline{s_0}^{\leq}(yes + no) + s.s_0.\overline{s_1}(yes + no)$ with $s = s_0.s_1$. To see this, consider that the traces up to length $|s|$ that do not cause a rejection of the trace s are the ones that do not cause a rejection of its

prefix s_0 and the ones that start with s_0 but do not cause the rejection of its continuation s_1. Thus we have:

$$x + s.x + s.\overline{s_0}^{\leq}(yes + no) + s.s_0.\sum_{a \in Act} a.(yes + no) + s.s_0.\overline{s_1}(yes + no)$$

$$= x + s.\overline{s_0}^{\leq}(yes + no) + s.s_0.\sum_{a \in Act} a.(yes + no) + s.s_0.\overline{s_1}(yes + no)\ .$$

Now we have again that the summand $s.s_0.\sum_{a \in Act} a.(yes + no)$ accepts and rejects a prefix of the whole summand $s.s_0.\overline{s_1}(yes + no)$ and therefore we can omit the latter. This gives us the equation:

$$x + s.x + s.\overline{s_0}^{\leq}(yes + no) + s.s_0.\sum_{a \in Act} a.(yes + no) =$$

$$x + s.\overline{s_0}^{\leq}(yes + no) + s.s_0.\sum_{a \in Act} a.(yes + no)\ ,$$

which can be rewritten using our notation as

$$x + s.x + s.\overline{s_0}(yes + no) = x + s.\overline{s_0}(yes + no)\ ,$$

giving us the target equation.

Having presented the proof for the first family of equations in detail we give a short description for the rest. For the equations (2) and (3) it suffices to use the congruence rule for $+$ with the equations $yes = yes$ and $no = no$ respectively and then simplify the equations by using the distribution axiom for $+$. For the latter equation (4) it is enough to instantiate the prefix s_0 in the the family of equations (1) as the empty string ε. This is, of course, allowed since the empty string is a prefix of any string. □

Now that we have discussed the family of axioms $\mathcal{O}$, we proceed to use them in defining a notion of reduced normal form that is suitable for monitors over a finite action set.

Definition 9. *A **finite-action-set reduced normal form** is a term*

$$m = \sum_{a \in A} a.m_a + \sum_{i \in I} x_i\ [+yes]\ [+no]$$

where each m_a is different from end and if $v \in \{yes, no\}$ is a summand of m then each m_a is v-free, and in reduced normal form. If both yes and no are summands of m then m is equal to $yes + no$. In addition for every trace s, if there exists a k such that for all the traces s_0

$$\overline{s}^{(k)}(yes + no) \xrightarrow{s_0} yes + no,\ \textit{implies: } m \stackrel{s_0}{\Longrightarrow} yes \textit{ and } m \stackrel{s_0}{\Longrightarrow} no$$

then $m \stackrel{s}{\not\rightarrow} x_i + m'$ for all $i \in I$ and m'.

In order to use the above form of the monitors in Mon_F we need to prove that any term can be rewritten in a reduced normal form using the axioms in $\mathcal{E}'_{v,f}$. Before doing so we will prove the following useful lemma, which only uses axioms form $\mathcal{E}_v$.

Lemma 14. *For a monitor $m \in Mon_F$:*

- *if $m \stackrel{s}{\Rightarrow} yes$ then $\mathcal{E}_v \vdash m = m + s.yes$ and*
- *if $m \stackrel{s}{\Rightarrow} no$ then $\mathcal{E}_v \vdash m = m + s.no$.*

Proof. We prove both statements by induction on the length of the trace s and limit ourselves to presenting the proof for the first one.

- If s is the empty trace, then m accepts the empty trace. Therefore it must contain a *yes* syntactic summand and we are done.
- Assume now that $s = a.s'$. Then $m \stackrel{a}{\rightarrow} m_a \stackrel{s'}{\rightarrow} yes$ for some m_a. By induction $\mathcal{E}'_{v,f} \vdash m_a = m_a + s'.yes$.

  Now,

$$\begin{aligned}\mathcal{E}'_{v,f} \vdash m = m + a.m_a = m + a.(m_a + s'.yes) = m + a.m_a + a.s'.yes \\ = m + a.s'.yes\end{aligned}$$

□

We will also need a similar result, this time involving syntactic summands that contain occurrences of variables.

Lemma 15. *For a monitor $m \in Mon_F$, where m is in normal form for open terms, if $m \stackrel{s}{\rightarrow} x + m_s$ then $\mathcal{E}_v \vdash m = m' + s.x$ where $m' \stackrel{s}{\not\rightarrow} x + m''$ for every m''.*

Proof. We prove the claim by induction on the length of the trace s.

- If s is the empty trace then $m \stackrel{\varepsilon}{\rightarrow} x + m_s = m$. This means that x is a summand of m. Since m is in normal form, m_s does not have x as a summand and we are done.
- Assume now that $s = a.s'$. Since m is in normal form and $m \stackrel{a.s'}{\longrightarrow} s + m'$, we have that $m = m' + a.m_a$ for some $m' \stackrel{a}{\not\rightarrow}$ and m_a in formal form such that $m_a \stackrel{s'}{\rightarrow} x + m'$. By the induction hypothesis, $\mathcal{E}_v \vdash m_a = m'_a + s'.x$ where $m'_a \stackrel{s'}{\not\rightarrow} x + m'_{s'}$ for every $m'_{s'}$.

  Therefore we have:

$$m = m' + a.m_a = m + a.(m'_a + s'.x) = m' + a.m'_a + a.s'.x \ ,$$

  and since $m'_a \stackrel{s'}{\not\rightarrow} x + m'_{s'}$ for every $m'_{s'}$ and $m' \stackrel{a}{\not\rightarrow}$ we have that $m = s.x + m_{rest}$ with $m_{rest} \stackrel{s}{\not\rightarrow} x + m''$ for every m'' and we are done. □

Lemma 16. *Each open monitor $m \in Mon_F$, is provably equal to a reduced normal form using $\mathcal{E}'_{v,f}$.*

Proof. From Lemma 8 we can start from a monitor m already in open normal form, as given in Definition 6. Therefore we have the following cases:

- $m = yes + no$.
- $m = yes + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$, where each m_a is yes-free.
- $m = no + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$, where each m_a is no-free.
- $m = \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$.

We begin our analysis from the second case. A similar analysis can be applied to the third one and the fourth one follows by a simpler version of the same inductive argument. We have therefore a monitor $m = yes + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i$. The extra claim for these reduced normal forms is that if for some trace s, and a k_0, for all traces s_0,

$$\overline{s}^{(k_0)}(yes + no) \xrightarrow{s_0} yes + no, \text{ implies: } m \stackrel{s_0}{\Longrightarrow} yes \text{ and } m \stackrel{s_0}{\Longrightarrow} no$$

then $m \stackrel{s}{\not\rightarrow} x_i + m_x$ for all $i \in I$ and m_x. In order to prove this extra constraint we assume the premise is true. We will show that we can reduce m to m_{red} with $\mathcal{E}_{fin} \vdash m = m_{red}$ and $m_{red} \stackrel{s}{\not\rightarrow} x_i + m_x$ for every $i \in I$ and every m_x.

Since m accepts and rejects all the traces that $\overline{s}^{(k_0)}(yes + no)$ accepts and rejects, we have that $m \simeq m + m'$ and that $m' \xrightarrow{s_0} yes + no$ for all of the traces s_0 that $\overline{s}^{(k_0)}(yes + no) \xrightarrow{s_0} yes + no$. We call this set of traces $\mathcal{S}$ which is finite since k_0 is fixed. Therefore by Lemma 14 we have that $\mathcal{E}'_{v,f} \vdash m = m + \sum_{s_0 \in \mathcal{S}} s_0.(yes + no)$. Since the term $\sum_{s_0 \in \mathcal{S}} s_0.(yes + no)$ is verdict equivalent to $\overline{s}^{(k_0)}(yes + no)$ and both terms are closed, we have that by Theorem 2, $\mathcal{E}_v \vdash \sum_{s_0 \in \mathcal{S}} s_0.(yes + no) = \overline{s}^{(k_0)}(yes + no)$. Therefore $\mathcal{E}'_{v,f} \vdash m = m + \overline{s}^{(k_0)}(yes + no)$, which means

$$\mathcal{E}'_{v,f} \vdash m = yes + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i + \overline{s}^{(k_0)}(yes + no).$$

For the same monitor m we now want to argue that if $m \xrightarrow{s} x_i$ for one of the variables in $\{x_i \mid i \in I\}$ then we can eliminate this occurrence.

Since m is in reduced normal form we have by Lemma 15 that $m = m' + s.x_i$ where $m' \stackrel{s}{\not\to} x_i$. Additionally we have shown that $m = m + \overline{s}^{(k_0)}(yes + no)$ which implies $m = m' + s.x_i + \overline{s}^{(k_0)}(yes + no)$ with $m' \stackrel{s}{\not\to} x_i$. Since x_i is one of the variables that appear as summands of m we can successfully apply the axiom $O2_{s,k_0}$ for each variable and we have the that indeed m reduces to a monitor m_{red} such that $m_{red} \stackrel{s}{\not\to} x_i + m_{x_i}$ for every $i \in I$ and every m_{x_i}. □

Lemma 17. *If monitor $m \in Mon_F$, with $|Act| \geq 2$ is in reduced normal form and contains an x summand and $m \stackrel{s}{\to} x + m'$ for some m' then there is at least one trace s_{bad} such that for every k,*

$$\overline{s}^{(k)}(yes + no) \stackrel{s_{bad}}{\Longrightarrow} yes \text{ and } \overline{s}^{(k)}(yes + no) \stackrel{s_{bad}}{\Longrightarrow} no$$

but

$$m \stackrel{s_{bad}}{\not\Longrightarrow} yes \text{ or } m \stackrel{s_{bad}}{\not\Longrightarrow} no.$$

Proof. We can easily show that for each k there exists an s_k such that $\overline{s}^{(k)}(yes + no) \stackrel{s_k}{\longrightarrow} yes + no$ but $m \stackrel{s_k}{\not\Rightarrow} yes$ or $m \stackrel{s_k}{\not\Rightarrow} no$. This follows since if this were not the case then for some k_0, no such trace s_{k_0} exists. Thus the monitor would contain a summand $m' \simeq \overline{s}^{(k_0)}(yes + no)$ for this k_0 and still it would be able to perform the transition $m \stackrel{s}{\to} x + m'$ which contradicts the assumption that m is in reduced normal form.

We will now show that one trace s_{bad} suffices for all k. To that end, consider the term $\overline{s}^{(1)}(yes + no)$. If there is an s_1, which is not a prefix of ss and $m \stackrel{s_1}{\not\Rightarrow} yes$ or $m \stackrel{s_1}{\not\Rightarrow} no$ then for $s_{bad} = s_1$ we have that for all k, $\overline{s}^{(k)}(yes + no) \stackrel{s_{bad}}{\Longrightarrow} yes$ and $\overline{s}^{(k)}(yes + no) \stackrel{s_{bad}}{\Longrightarrow} no$ and we are done. If this is not the case and since the trace s_1 is guaranteed to exist (by the previous paragraph) then it must be an extension of ss. Again if s_1 is not prefix of sss then again for $s_{bad} = s_1$ we have the necessary conclusion.

Otherwise $m \stackrel{s_1}{\not\Rightarrow} yes$ or $m \stackrel{s_1}{\not\Rightarrow} no$ for the trace $s_1 = ssa$, where a is the first action of s. Therefore by the definition of $\overline{s}^{(k)}(yes + no)$ we have that for all s_b such that $\overline{s}^{(k)}(yes + no) \stackrel{s_b}{\longrightarrow} yes + no$ and $k > 1$ we have that $m \stackrel{s_b}{\not\Rightarrow} yes$ or $m \stackrel{s_b}{\not\Rightarrow} no$. This allows us to look for an s_{bad} which will also cover the case $k = 1$ in larger terms.

We then apply the same reasoning for $k = 2, \ldots$ up to a certain k_b. If at any point in the process we encounter a trace s_i which fulfills our premise then we can stop. We are just left to show that this process will eventually terminate.

This can be shown as follows. Recall that every monitor m has a finite depth $depth(m)$ (see Def. 2). We now take a k_b large enough so that $s^{k_b} > depth(m)$. If the iterative procedure described above reaches this k_b we have that $m \stackrel{s_{bad}}{\not\Longrightarrow} yes$ or $m \stackrel{s_{bad}}{\not\Longrightarrow} no$ for the trace $s^{k_b+1}a$ where a is the first action of s. However since

the depth of the monitor m is smaller that the length of this trace we also have that the monitor cannot accept or reject any of its extensions.

Therefore for the extension $s_{bad} = s^{k_b+1}ac$ where c is not the second action of s we have that for all $k > k_b$, $\overline{s}^{(k)}(yes + no) \stackrel{s_{bad}}{\Longrightarrow} yes$ and $\overline{s}^{(k)}(yes + no) \stackrel{s_{bad}}{\Longrightarrow} no$ while m
$not \stackrel{s_{bad}}{\Longrightarrow} yes$ or $m \stackrel{s_{bad}}{\not\Longrightarrow} no$. Additionally since the iterative procedure we described above reached this k_b we have that for all $i \leq j \leq k_b$, it is true that $\overline{s}^{(k)}(yes + no) \stackrel{s_j}{\Rightarrow} yes$ and $\overline{s}^{(k)}(yes + no) \stackrel{s_j}{\Rightarrow} no$, which concludes the proof. □

The two lemmata above play a key role in the completeness proof we will present now.

We distinguish two cases separately, namely when $|Act| \geq 2$ and when Act is a singleton. This is necessary because equations such as $x = x + a.x$ are only sound when $Act = \{a\}$. For the proof when $|Act| \geq 2$ it is necessary to utilize at least two actions $a, b \in Act$, which is the reason why when only one action is available new cases arise.

Action set with at least two actions. We have already shown the soundness of the axiom system $\mathcal{E}'_{v,f}$. We now proceed to show completeness.

For each such completeness theorem we follow a similar general strategy in order to prove that two arbitrary verdict equivalent monitors have identical reduced normal forms. To that end, we prove that they have identical variables as summands, that the sets of initial actions that each one can perform are equal and that after a common action they reach monitors that are also verdict equivalent. Unfortunately, for a finite set of actions, we were not able to define a substitution that would cover all the three above-mentioned steps like we did when the set of actions was infinite. We therefore adopted a proof strategy that focuses on each part of the proof separately.

Theorem 5. *$\mathcal{E}'_{v,f}$ is complete for open terms for finite Act with $|Act| \geq 2$. That is, if $m \simeq n$ then $\mathcal{E}'_{v,f} \vdash m = n$.*

Proof. By Lemma 16 we may assume that m and m are in reduced normal form. We prove the claim by induction on the sum of the sizes of m and n, and proceed with a case analysis on the form m may have.

In the case where m contains both a *yes* and a *no* summand then both m and n must be equal to $yes + no$ as they are in reduced normal form.

Assume now that

$$m = yes + \sum_{a \in A} a.m_a + \sum_{i \in I} x_i \ ,$$

where $\{x_i \mid i \in I\}$ is the set of variables occurring as summands of m and each m_a is *yes*-free and different from *end* (as a reduced normal form). Since $\sigma(m)$ accepts ε for each σ and $m \simeq n$, monitor n is bound to have a similar form since it must contain the verdict *yes* as a summand (but not a *no* one). Therefore:

$$n = yes + \sum_{b \in B} b.n_b + \sum_{j \in J} y_j$$

and we need to show that there is a way to apply our axioms to show that monitor n is provably equal to m.

We start by proving that $\{x_i \mid i \in I\} = \{y_j \mid j \in J\}$. By symmetry, it suffices to show that $\{x_i \mid i \in I\} \subseteq \{y_j \mid j \in J\}$. To this end, assume $x \in \{x_i \mid i \in I\}$. Consider the substitution σ mapping x to no and every other variable to end, i.e:

$$\sigma(y) = \begin{cases} no, & \text{if } y = x \\ end, & \text{otherwise.} \end{cases}$$

Then, $\sigma(m)$ rejects the empty trace ε. Since $\sigma(m) \simeq \sigma(n)$, we have that $\sigma(n)$ must also reject ε. By the form of n and the definition of σ, this is only possible if n has x as a summand, and we are done. Therefore the set of variables of m is a subset of the variables of n.

Next, we prove that the action sets A, B are identical. Assume that $a \in A$. Since Act contains at least two actions, there is some action $b \neq a$. Consider the substitution σ_1 defined by $\sigma_1(x) = b.no$ for each $x \in Var$. Since $a \in A$ and m_a is yes-free and different from end, it is easy to see that there exists an $s \in Act^*$ such that $as \in L_r(\sigma_1(m))$. Since $m \simeq n$ we have that $\sigma_1(m) \simeq \sigma_1(n)$ and therefore $\sigma_1(n)$ must also reject as. By the form of n and the definition of σ, this is only possible if $n \xrightarrow{a} n_a$ for some n_a and therefore $a \in B$. Hence, $A \subseteq B$ and the claim follows by symmetry.

For the final part of the proof we must show that $m_a \simeq n_a$ for each $a \in A$, which is enough to complete the proof, by the induction hypothesis. Towards a contradiction we will assume that the two monitors m_a, n_a are not verdict equivalent. Therefore there exists a substitution σ_0 that separates them, that is without loss of generality, there is a trace s_0 such that $s_0 \in L_r(\sigma_0(m_a)), s_0 \notin L_r(\sigma_0(n_a))$ or there is some $s_0 \in L_a(\sigma_0(m_a)), s_0 \notin L_a(\sigma_0(n_a))$.

We will analyze first the case of rejection of the string s_0. The substitution σ_0 must be a closed one for m_a, n_a i.e. it must map to a closed monitor all variables in $(Var(m_a) \cup Var(n_a))$. We will use this substitution to create a new one σ_{bad} that would also separate the original monitors m, n.

The first step towards this is:

$$\sigma_{bad}(x) = \begin{cases} end, \text{ if } x \in Var(m) \setminus (Var(m_a) \cup Var(n_a)), \\ \sigma_0(x), \ otherwise. \end{cases}$$

Now since $s_0 \in L_r(\sigma_0(m_a))$ and $\sigma_{bad}(m_a) = \sigma_0(m_a)$ we also know that $a.s_0 \in L_r(\sigma_{bad}(a.m_a))$. Our aim is to show that $a.s_0 \notin L_r(\sigma_{bad}(n))$. Following the definition $\sigma_{bad}(n_a) = \sigma_0(n_a)$ and therefore $s_0 \notin L_r(\sigma_{bad}(n_a))$.

Hence, the only way for $\sigma_{bad}(n)$ to reject $a.s_0$, like $\sigma_{bad}(m)$ does, is if it was rejected by the mapping of one the variables contained in the set $\{x_i \mid i \in I\}$.

It is useful to make here apparent that in order for $\sigma_{bad}(n)$ to reject $a.s_0$, it must do so completely independently of the summand $\sigma_{bad}(a.n_a)$, since the latter cannot reject any of the prefixes of $a.s_0$ as well. Even in the case where s_0 starts with a, and $\sigma_0(n_a)$ rejects some $a.s_1.s_2.\ldots.s_{n-i}$ it would still be impossible for $\sigma_0(n_0)$ to reject $a.s_0$ since the assumption that $a.s_0 = a.a.s_1.s_2.\ldots.s_{n-1}$

would automatically imply that $\sigma_0(n_a)$ rejects some prefix of s_0 which is a contradiction.

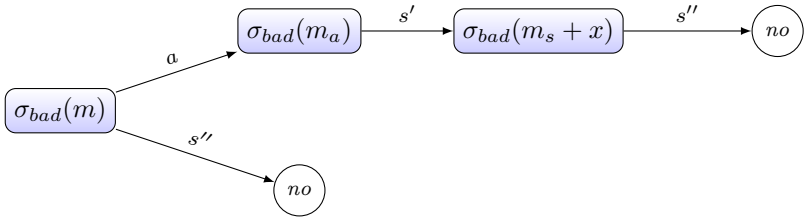

Figure 1: Transitions the monitor $\sigma_{bad}(m)$ can perform

By the definition of σ_{bad}, the variables that did not appear at all in n_a or m_a were mapped to *end* and therefore cannot reject any string. Therefore the only way for n to reject $a.s_0$ is for one of the variables appearing in $Var(n_a) \cup Var(m_a)$ to have been mapped to a closed term that can reject $a.s_0$. (Note that this does not contradict the fact that $\sigma_{bad}(n_a)$ does not reject s_0). Therefore there is at least one $x_0 \in Var(m_a) \cup Var(n_a)$ and $x_0 \in \{x_i \mid i \in I\}$ such that $as_0 \in L_r(\sigma_{bad}(x_0))$.

This leads to the case were m, n reject a prefix of as_0 because of the mapping of x_0. However this implies that we have the following situation:

$$m = yes + x_0 + a.m_a + \sum_{b \in A \setminus \{a\}} b.m_b + \sum_{i \in I \setminus \{0\}} x_i \simeq$$

$$yes + x_0 + a.n_a + \sum_{b \in A \setminus \{a\}} b.n_b + \sum_{i \in I \setminus \{0\}} x_i = n$$

and that the monitor m_a can perform the transitions: $m_a \xrightarrow{s'} m'_a + x_0$ and the monitor $\sigma_0(x_0) = \sigma_{bad}(x_0)$ respectively can perform the transitions: $\sigma_{bad}(x_0) \xrightarrow{s''} no$, where s' is a prefix of s_0 (i.e. $s_0 = s'.s''$) and in addition $n_a \overset{s'}{\not\longrightarrow} x + n'$ for any n'. This means respectively that $m \xrightarrow{as'} m'_a + x_0$ and $\sigma_{bad}(m'_a + x_0) \xrightarrow{s''} no$.

By Lemma 17 we have that there exists at least one trace s_b such that $m \overset{s_b}{\not\Rightarrow} yes$ or $m \overset{s_b}{\not\Rightarrow} no$ but $s_b \in L_r(\overline{as'}^{(k)}(yes + no))$ for all $k \geq 0$. Since m contains a *yes* summand we have that it must be the case that $m \overset{s_b}{\not\Rightarrow} no$. We now, further modify σ_{bad} to map the variable x_0 to $s_b.no$ and any other variable $y \neq x_0$ to *end*. We have then that s_b and $as'.s_b \in L_r(\sigma_{bad}(m))$. In addition $s_b \in L_r(\sigma_{bad}(n))$. However the traces that are rejected by the term $\overline{as'}^{(k)}$, by definition, are exactly the traces such that their rejection does not cause a rejection of the as' trace. This means that under the modified substitution σ_{bad}, monitor n *cannot* reject the trace $as'.s_b$. This deems the monitors m, n not verdict equivalent, which contradicts our assumption. We conclude then that the rejection set of m_a is equal to the rejection set of n_a for each $a \in A$.

It remains to show that m_a and n_a also have identical acceptance sets. Towards a contradiction, assume they do not and take a trace s that under some substitution σ_0 separates them, i.e. $s \in L_a(\sigma_0(m_a))$ and $s \notin L_a(\sigma_0(n_a))$. In addition, assume that s is of minimum length, meaning that no prefix of s (under any substitution) has the property of separating the acceptance sets of m_a and n_a. This fact in addition to m_a and n_a being *yes*-free (as a result of m and n being in reduced normal form) means that the acceptance of s by m_a is the result of a variable x occurring in m_a as $m_a \xrightarrow{s} x + m'$ for some m'. Since however the assumption is that $s \notin L_a(\sigma_0(n_a))$ we have that $n_a \not\xrightarrow{s} x + n'$ for any n'. We know that this is exactly the case since if the variable x occurred earlier in m_a then by mapping it to *yes* we would have a shorter trace being accepted by $\sigma_0(m_a)$ but not $\sigma_0(n_a)$.

We are sure now that monitor $\sigma_0(n_a)$ cannot perform the transition $\sigma_0(n_a) \stackrel{s}{\Longrightarrow} \mathit{yes}$, which means that not only it does not arrive at the variable x after reading the trace s, but also does not arrive to the *yes* verdict for any of its prefixes (say s') as that would imply that it can reach the *yes* verdict for s as well.

Finally, by n being in reduced normal form, and by m_a not arriving at a *no* verdict for any of the prefixes s' of s (as this would mean that if becomes a *no* and therefore cannot perform the transitions $m_a \xrightarrow{s} x$) we know that n_a does not arrive to the *no* verdict after reading the trace s or any of its prefixes either.

Given all of the above we can now construct the substitution σ_{bad} that would separate the rejection sets of n_a, m_a which is enough to prove the contradiction as the case where such a substitution exists and separates the rejection sets of the two sub-monitors has already been covered. The situation we have at hand is as follows:

Monitor $\sigma_0(m_a)$ can arrive to the verdict *yes* after reading the trace s while $\sigma_0(n_a)$ cannot and also neither n_a nor m_a can produce a *no* verdict for the trace s. Therefore if we switch the mapping of x to *no* in σ' and the verdicts of all other variables that where mapped to a *no* verdict to *end* we have produced a substitution that causes s to be rejected by $\sigma'(m_a)$ but not from $\sigma'(n_a)$. By utilizing our previous construction there exists another one that separates the monitors n, m as well which is a contradiction.

We have concluded then that the $L_a(m_a) = L_a(n_a)$ and $L_r(m_a) = L_r(n_a)$ which means that they are verdict equivalent. Therefore we can apply the inductive hypothesis and have that $\mathcal{E}'_{v,f} \vdash m_a = n_a$. Using now congruence rules we have that $\mathcal{E}'_{v,f} \vdash m = n$. All other possible forms of monitors m, n are sub-cases that the relative analysis can be applied symmetrically and therefore they are omitted. □

Singleton Action Set. We proceed now with the analysis of the completeness result when $Act = \{a\}$.

As we mentioned earlier, when a is the only action, the equation

$$(V_1) \quad x = x + a.x$$

is sound, but cannot be proved from the equations in $\mathcal{E}'_{v,f}$ over $\{a\}$. Indeed, unlike V_1, all the equations in E_v are sound regardless of the cardinality of the action set and those in the family $\mathcal{O}$ introduce subterms of the form $yes + no$, which can never be removed in equational derivations.

Theorem 6. *The finite axiom system $\mathcal{E}'_{v,1} = \mathcal{E}'_v \cup \{V_1\}$ is complete for verdict equivalence over open monitors when $Act = \{a\}$. That is, if $m \simeq n$ then $\mathcal{E}'_{v,1} \vdash m = n$. Hence, verdict equivalence is finitely based when $Act = \{a\}$.*

Proof. Before we start the main proof we note that the new axiom V_1 can prove the equation $x = a^n.x + x$ for each $n \geq 0$. This is done as follows: if $n = 0$ then this is the axiom $A3$. Assume we can prove that equation for n. Then we can show it for $n + 1$ thus:

$$x \overset{\mathrm{V_1}}{=} x + a.x \overset{\mathrm{I.H.}}{=} x + a.(a^n.x + x) \overset{\mathrm{D_a}}{=} x + a.x + a^{n+1}.x \overset{\mathrm{V_1}}{=} x + a^{n+1}.x \quad .$$

Note here that this means that $\mathcal{E}'_v \cup \{V_1\}$ proves all the equations in $\mathcal{O}$ over $\{a\}$, which means that even though $\mathcal{E}'_v \cup \{V_1\}$ is finite, it can prove the infinite family $\mathcal{E}'_{v,f}$ over $\{a\}$.

Let $m \simeq n$. By Lemma 16, we can assume that m and n are in reduced normal form. We will present the argument only for the case where $m = yes + a.m_a + \sum_{i \in I} x_i$, where each m_a is *yes*-free, as every other case is either trivial or a sub-case of this one.

By following the reasoning of previous proofs, we have that $n = yes\ [+a.n_a] + \sum_{i \in I} x_i$.

Let us first consider the case that $a.n_a$ is not a summand of n. (Note that this is possible, as witnessed by axiom V_1.) That is

$$m = yes + a.m_a + \sum_{i \in I} x_i \simeq yes + \sum_{i \in I} x_i = n \quad .$$

Observe that, for each $s \in Act^*$, we have $m_a \overset{s}{\not\Rightarrow} no$. Indeed, $m_a \overset{s}{\Rightarrow} no$ would imply that m and n are not verdict equivalent under the substitution $\sigma_{end}(x) = end$ for all x. This means that m_a is both *yes*- and *no*-free. Moreover, note that the set of variables occurring in m_a is included in $\{x_i \mid i \in I\}$. To see this, assume that x occurs in m_a, but is not contained in $\{x_i \mid i \in I\}$. Consider the substitution that maps x to *no* and all the other variables to *end*. Again, we have that m rejects some trace starting with a while n cannot reject any trace, which contradicts our assumption that $m \simeq n$.

For each monitor m', we define $\mathcal{V}(m')$ as the set of pairs (s, x) such that $m' \overset{s}{\to} x + m''$ for some m''. By structural induction on m' and Lemma 15, one can easily prove that, when m' is *yes*- and *no*-free, $\mathcal{E}_v$ proves $m' = \sum_{(s,x) \in \mathcal{V}(m')} s.x$.

Therefore $m = yes + a.\sum_{(s,x) \in \mathcal{V}(m_a)} s.x + \sum_{i \in I} x_i$. Since the only available action in Act is a and the variables occurring in m_a also occur in $\{x_i \mid i \in I\}$,

we have that by applying the equations we proved earlier by using axiom V_1 we can prove $m = yes + \sum_{i \in I} x_i = n$, and we are done.

Assume now that $a.n_a$ is a summand of n. We proceed to prove that that $m_a \simeq n_a$. In this case we have

$$m_a = \sum_{(s,x) \in \mathcal{V}(m_a)} s.x\ [+a^h.no] \text{ and } n_a = \sum_{(s,x) \in \mathcal{V}(n_a)} s.x\ [+a^k.no]\ ,$$

for some h, k.

By mapping all variables to *end* we can see that $h = k$. Additionally, for each variable s and by using the axiom V_1 we can reduce both of the above summations so that only the shortest s leading to x is kept. By Lemma 10, we have that, for each variable, this s is identical for both sides of the equality $m \simeq n$ and we are done. □

4.2.2. Completeness of ω-verdict equivalence

This section presents a complete axiomatization for ω-verdict equivalence over Mon_F. We have already presented the necessary axioms that capture ω-verdict equivalence over closed terms, as well as the necessary ones to capture equivalence of terms that include variables. We will show here that the combination of the two axiom systems is enough for completeness of ω-verdict equivalence over open terms and there is no need for extra axioms to be added. First we look at the case for a singleton action set, i.e. $Act = \{a\}$. In this case, the equation

$$(\mathbf{V1}_\omega)\ x = a.x$$

is sound and we therefore we can shrink the axiom system to:

$$\mathcal{E}'_{\omega,1} = \{A1 - A4\} \cup \{V1_\omega\} \cup \{O1\},$$

for which we prove:

Theorem 7. *$\mathcal{E}'_{\omega,1}$ is complete for ω-verdict equivalence for open terms for a finite Act, with $|Act| = 1$. That is, if $m \simeq_\omega n$ then $\mathcal{E}'_{\omega,1} \vdash m = n$.*

Proof. The proof of the above follows easily since, by using those equations, every term can be proved equal to one of the form $\sum_{i \in I} x_i\ [+yes]\ [+no]$, where I is empty if both *yes* and *no* are summands, and two terms of that form are ω-verdict equivalent iff they are equal modulo $A1 - A4$. Note that, in this case, there are only four congruence classes of terms, namely the ones asymptotically equivalent to subsets of *yes* and *no*, so the quotient algebra is very small and equationally well behaved. □

In the case where there the action set contains more than one action but is still finite we have a more interesting situation. We therefore define:

$$\mathcal{E}'_{\omega,f} = \mathcal{E}_\omega \cup \mathcal{E}'_{v,f}\ ,$$

for which we prove:

Theorem 8. *$\mathcal{E}'_{\omega,f}$ is complete for ω-verdict equivalence over open terms when Act is finite and $|Act| \geq 2$. That is, if $m \simeq_\omega n$ then $\mathcal{E}'_{\omega,f} \vdash m = n$.*

The rest of this section is devoted to the proof of the above theorem. We start by showing a lemma that tells us that if two monitors are ω-verdict equivalent then they can only disagree on finitely many finite traces.

Lemma 18. *For two monitors in Mon_F, we have that $m \simeq_\omega n$ if and only if, for any substitution σ, the set*

$$\begin{aligned}\mathcal{S}_{m,n,\sigma} = (L_a(\sigma(m)) \setminus L_a(\sigma(n))) \cup (L_r(\sigma(m)) \setminus L_r(\sigma(n))) \\ \cup (L_a(\sigma(n)) \setminus L_a(\sigma(m))) \cup (L_r(\sigma(n)) \setminus L_r(\sigma(m)))\end{aligned}$$

is finite.

Proof. We prove both implications separately by establishing their contrapositive statements. For the implication from left to right, assume that $\mathcal{S}_{m,n,\sigma}$ is infinite. It follows that there are some σ and trace s such that $s \in \mathcal{S}_{m,n,\sigma}$ with

$$|s| > max\{depth(\sigma(m)), depth(\sigma(n))\}.$$

Assume, without loss of generality, that $\sigma(m)$ accepts s, but $\sigma(n)$ does not. Let $a \in Act$. Then sa^ω is in $L_a(\sigma(m)) \cdot Act^\omega$. We claim that sa^ω is not in $L_a(\sigma(n)) \cdot Act^\omega$. Indeed, $\sigma(n)$ does not accept any prefix of s, since it does not accept s itself, and it does not accept sa^i for any $i \geq 0$ because $|s| > depth(\sigma(n))$. For the implication from right to left, assume, without loss of generality, that there are some substitution sigma and some $t \in Act^{\{}omega$ such that t is in $L_a(\sigma(m)) \cdot Act^\omega$, but not in $L_a(\sigma(n)) \cdot Act^\omega$. Since t is in $L_a(\sigma(m)) \cdot Act^\omega$, we have that there are some $s \in L_a(\sigma(m))$ and u in Act^ω such that $t = su$. It follows that $ss' \in L_a(\sigma(m))$ for each finite prefix s' of u, but none of the ss' is contained in $L_a(\sigma(n))$. Therefore, $\mathcal{S}_{m,n,\sigma}$ is infinite, and we are done. □

We are now ready to present the proof of the main theorem of this section (Theorem 8).

Proof. By Lemma 16 we may assume without loss of generality that the monitors m and n are in finite-action-set reduced normal form (Definition 9).

We proceed by a case analysis on the form m and n might have and by induction on the sum of the sizes of m and n.

- Assume that $m = yes + no \simeq_\omega \sum_{a \in A} a.n_a + \sum_{j \in J} y_j = n$. First of all, note that $A = Act$. Indeed, assume $a \notin A$. Then, under a substitution that maps every variable to *end*, all infinite traces starting with a are neither accepted nor rejected by n since n cannot take an a transition and it also does not accept and reject ε. However all infinite traces (including

those starting from a) are both accepted and rejected by m, which is a contradiction as we have assumed that the two monitors are ω-verdict equivalent.

Moreover, it is not hard to see that $n_a \simeq_\omega yes + no$ holds for each $a \in Act$. By the induction hypothesis, $\mathcal{E}'_{\omega,f}$ proves $n_a = yes + no$, for each $a \in Act$. Therefore,

$$\mathcal{E}'_{\omega,f} \vdash n = \sum_{a\in Act} a.(yes + no) + \sum_{j\in J} y_j \overset{\mathrm{D_a}}{=} \sum_{a\in Act} a.yes + \sum_{a\in Act} a.no + \sum_{j\in J} y_j$$

$$\overset{\mathrm{Y_\omega,N_\omega}}{=} yes + no + \sum_{j\in J} y_j \overset{\mathrm{O1}}{=} yes + no \quad ,$$

and we are done.

- Now, we assume that $m = yes + no \simeq_\omega \sum_{a\in A} a.n_a + \sum_{j\in J} y_j + yes = n$, with each n_a being *yes*- and *end*-free. As above $A = Act$. Moreover, for each $a \in Act$, $L_r(n_a) \cdot Act^\omega = Act^\omega$. Following the same argument as above only for the *no* verdict we conclude that

$$\mathcal{E}'_{\omega,f} \vdash n = yes + \sum_{a\in Act} a.no + \sum_{j\in J} y_j = yes + no = m.$$

- The case $m = yes + no \simeq_\omega \sum_{a\in A} a.n_a + \sum_{j\in J} y_j + no = n$ is symmetrical to the previous one.

- The final case whose proof we present in detail is when

$$m = yes + \sum_{a\in A} a.m_a + \sum_{i\in I} x_i \simeq_\omega \sum_{b\in B} b.n_b + \sum_{j\in J} y_j \ [+yes] \ [+no] = n \ ,$$

where each side is in reduced normal form. To deal with this case, we note, first of all, that by mimicking the argument in the first case of the proof, we can prove that $\mathcal{E}'_{\omega,f} \vdash n = yes + \sum_{b\in B'} b.n'_b + \sum_{j\in J} y_j$, where now each n'_b is *yes*-free. By the same argument as for the verdict equivalence case (Proof of Theorem 5) and by defining the appropriate substitutions σ we can infer that $A = B'$ and $\{x_i \mid i \in I\} = \{y_j \mid j \in J\}$. In other words, we have:

$$m = yes + \sum_{a\in A} a.m_a + \sum_{i\in I} x_i \simeq_\omega yes + \sum_{a\in A} a.n'_a + \sum_{i\in I} x_i \quad ,$$

where m and n are in finite-action-set reduced normal form for open terms. It remains to show that under every substitution σ we have that $\sigma(m_a) \simeq_\omega$

$\sigma(n'_a)$ so that we can apply our induction hypothesis and complete the proof.

Towards a contradiction assume that this is not the case. Therefore there exists a substitution σ for which there is at least one infinite trace s such that, without loss of generality, $s \in L_r(\sigma(m_a)) \cdot Act^\omega$ but $s \notin L_r(\sigma(n'_a)) \cdot Act^\omega$ or $s \in L_a(\sigma(m_a)) \cdot Act^\omega$ but $s \notin L_a(\sigma(n'_a)) \cdot Act^\omega$. We examine first the case of the rejection sets. Since $\sigma(m_a)$ rejects the infinite trace s, there is some finite prefix s_0 of s that is rejected by $\sigma(m_a)$. Note that $\sigma(m_a)$ will also reject all the finite prefixes of s that extend s_0. On the other hand, $\sigma(n'_a)$ does not reject any of those because it does not reject s.

As we saw in the proof of Theorem 5 this substitution and any such trace s_0 can be modified to a new substitution σ' such that $\sigma'(m) \not\simeq \sigma'(n)$ and consequently m is not verdict equivalent to n. Specifically from the proof of Theorem 5 we have that:

- Under the substitution σ', all variables except x are mapped to end.
- $m_a \xrightarrow{s'} x + m'_a$ for some m'_a and a trace s' that is a prefix of s_0.
- $n'_a \not\xrightarrow{s'} x + n''_a$ for any n''_a
- The variable x is mapped to $s_b.no$ for a trace s_b such that m rejects the trace $as's_b$, but n does not.

By Lemma 18 we have that the only way m can be ω-verdict equivalent to n is if the number of traces they disagree on, under any substitution (including σ'), is finite. Since monitor m is ω-verdict equivalent to n, both monitors must disagree on finitely many extensions of $as's_b$. This however can be done only if m_a and n'_a also disagree on finitely many extensions of $s's_b$. This is because we have seen that under σ', only the variable x can contribute to the rejections sets of the monitors and it does so by being mapped to $s_b.no$. However, as s_b is not a prefix of $as's_b$ we know that also none of its extensions are prefixes of $as's_b$. Therefore the rejection of s_b does not cause the rejection of any of the prefixes and extensions of $as's_b$. This implies that the infinite trace s is only rejected by $\sigma'(m_a)$ but not $\sigma'(n'_a)$, which implies that the monitors m_a and n'_a still disagree on infinitely many extensions of s_0 under the new substitution σ' which is a contradiction.

It is now easy to see that for each $a \in A$ and for each substitution σ we have that $L_r(a.\sigma(m_a)) \cdot Act^\omega = L_r(a.\sigma(n'_a)) \cdot Act^\omega$ which implies $L_r(\sigma(m_a)) \cdot Act^\omega = L_r(\sigma(n'_a)) \cdot Act^\omega$. It remains to see that $L_a(\sigma(m_a)) \cdot Act^\omega = L_a(\sigma(n'_a)) \cdot Act^\omega$.

To this end, assume, towards a contradiction, that there exist a substitution σ and an infinite trace s such that $s \in L_a(\sigma(m_a)) \cdot Act^\omega$ but $s \notin L_a(\sigma(n'_a)) \cdot Act^\omega$. Following the argument for the rejection sets, we can infer that there is a finite trace s_0 accepted by $\sigma(m_a)$ but not by

$\sigma(n'_a)$. Again by using the proof of Theorem 5 , we can transform σ into a σ' that causes a disagreement over the rejection of a trace s'_0 for $\sigma(m_a)$ and $\sigma(n'_a)$ i.e. $s'_0 \in L_r(\sigma'(m_a))$ but $s'_0 \notin L_r(\sigma'(n'_a))$. This, in turn, means we can apply the same reasoning as before for the rejection of a trace to reach a contradiction, namely that m and n are not ω-verdict equivalent.

We can therefore conclude that $\sigma(m_a) \simeq_\omega \sigma(n'_a)$ under any substitution σ and therefore we can apply our induction hypothesis to obtain $\mathcal{E}'_{\omega,f} \vdash m_a = n'_a$. Using the congruence rules, we have $\mathcal{E}'_{\omega,f} \vdash m = n$, and we are done. □

Table 4 summarizes the equational axiom systems we have obtained.

5. A non-finite-axiomatizability result

Observe that the family of axioms $\mathcal{O} = \{O2_{s,k} \mid s \in Act^*, k \geq 0\}$, which is included in $\mathcal{E}'_{v,f}$, is infinite. Thus it is natural to wonder whether verdict equivalence has a finite equational axiomatization over Mon_F. In the remainder of this section, we will provide a negative answer to that question by showing that no finite subset of $\mathcal{E}'_{v,f}$ is enough to prove all the equations in $\mathcal{O}$.

Intuitively, the proof of the above claim proceeds as follows. Let $\mathcal{E}$ be an arbitrary finite subset of $\mathcal{E}'_{v,f}$. First of all, we isolate a property of equations that is satisfied by all the equations that are provable from $\mathcal{E}$. We then show that there are equations in the family $\mathcal{O}$ that do not have the given property. This means that those equations are not provable from $\mathcal{E}$ and, therefore, that $\mathcal{E}$ cannot be complete for verdict equivalence.

An arbitrary finite axiom set vs. a finite subset of $\mathcal{E}'_{v,f}$. In Section 4.2.1, in Theorem 5, we proved that $\mathcal{E}'_{v,f}$ is complete for open terms over a finite action set modulo verdict equivalence. Therefore, without loss of generality, we can assume that this basis is in fact a subset of the equations in $\mathcal{E}'_{v,f}$. To see this, consider any sound equation that could be involved in an arbitrary axiom set. Since $\mathcal{E}'_{v,f}$ is complete this equation is derivable from it. In addition, since every proof is finite, there is a finite number of axioms of $\mathcal{E}'_{v,f}$ involved in this proof. Therefore, any finite family of equations is derivable from a finite subset of the equations in $\mathcal{E}'_{v,f}$. This means that if another finite family of equations was complete, there would also be a finite subset of equations from $\mathcal{E}'_{v,f}$ which would also be complete. From now on, when considering a finite equational basis we will always mean a subset of the equations in $\mathcal{E}'_{v,f}$.

We remind our readers that we assume that all axiom systems that we consider are closed under symmetry. This preserves finiteness and allows us to simplify our arguments, since the symmetry rule does not need to be used in equational proofs.

Definition 10 (**Notation**). *For a finite, non empty set of equations $\mathcal{E}$ we denote as $depth(\mathcal{E})$ the quantity:*

$$max\{depth(m) \mid m = n \in \mathcal{E}\}.$$

(A1) $x + y = y + x$	$(\mathbf{E_a})$ $a.end = end$ $(a \in Act)$
(A2) $x + (y + z) = (x + y) + z$	$(\mathbf{Y_a})$ $yes = yes + a.yes$ $(a \in Act)$
(A3) $x + x = x$	$(\mathbf{N_a})$ $no = no + a.no$ $(a \in Act)$
(A4) $x + end = x$	$(\mathbf{D_a})$ $a.(x + y) = a.x + a.y$ $(a \in Act)$

The axioms of $\mathcal{E}_v$, which are ground complete for $\simeq$ (Theorem 2).

$$(\mathbf{Y}_\omega)\ yes = \sum_{a \in Act} a.yes \qquad (\mathbf{N}_\omega)\ no = \sum_{a \in Act} a.no$$

The axiom system $\mathcal{E}_\omega = \mathcal{E}_v \cup \{Y_\omega, N_\omega\}$ is ground complete for $\simeq_\omega$ when Act is finite (Theorem 3).

$$(\mathbf{O1})\ yes + no = yes + no + x$$

The axiom system $\mathcal{E}'_v = \mathcal{E}_v \cup \{O1\}$ is complete for $\simeq$ when Act is infinite (Theorem 4).

$\mathcal{O} = \{O2_{s,k} \mid s \in Act^*, k \geq 0\}$ where

$$(\mathbf{O2_{s,k}})\ x + s.x + \overline{s}^{(k)}(yes + no) = x + \overline{s}^{(k)}(yes + no)$$

The axiom system $\mathcal{E}'_{v,f} = \mathcal{E}'_v \cup \mathcal{O}$ is complete for $\simeq$ when Act is finite and $|Act| \geq 2$ (Theorem 5).

$$(\mathbf{V_1})\ a.x + x = x$$

The axiom system $\mathcal{E}'_{v,1} = \mathcal{E}'_v \cup \{V_1\}$ is complete for $\simeq$ when $|Act| = 1$ (Theorem 6).

$$(\mathbf{V1}_\omega)\ x = a.x$$

The axiom system $\mathcal{E}'_{\omega,1} = \{A1, \ldots, A4, V1_\omega, O1\}$ is complete for $\simeq_\omega$ when $|Act| = 1$ (Theorem 7).

The axiom system $\mathcal{E}'_{\omega,f} = \mathcal{E}_\omega \cup \mathcal{E}'_{v,f}$ is complete for $\simeq_\omega$ when Act is finite and $|Act| \geq 2$ (Theorem 8).

Table 4: Our axiom systems

The depth of an axiom system turns out to be a very important aspect of it when proving open equations. We refer the reader to all the axioms we have defined so far (Figure 4) and particularly to the family $\mathcal{O}$. Take an instance of the family of equations $\mathcal{O}$, namely

$$x + a^k.x + \overline{a^k}^3(yes + no) \simeq x + \overline{a^k}^3(yes + no) \ ,$$

for some k. What we will focus on for equations like this one is the fact that every trace starting with s^k followed by any trace of length larger than $3k + 1$ (which is the depth of this equation), is both accepted and rejected by both sides of the equation for any closed substitution. This fact is exactly the intuition behind the property that we will use. We now proceed to formulate this property formally:

Lemma 19. *Let $\mathcal{E}$ be a finite subset of $\mathcal{E}'_{v,f}$ and let $m = n$ be an equation in $\mathcal{E}$. Assume that for some string s:*

- *$m \xrightarrow{s} m' + x$, for some monitor m' and variable x and*
- *$n \not\xrightarrow{s} n' + x$ for any n'.*

Then, for every trace of the form $s.s'$ where $|s'| \geq depth(\mathcal{E})$, we have that $ss' \in L_a(\sigma(m))$ and $ss' \in L_r(\sigma(m))$ for every substitution σ.

Proof. It suffices to examine each member of $\mathcal{E}'_{v,f}$ separately.

- Each axiom in $\mathcal{E}_v$ does not have any one-sided occurrence of a variable as the ones stated and therefore the lemma holds vacuously.
- For the axiom $O1$ we have that both sides accept and reject all traces for each σ and therefore the claim follows trivially.
- We are left to discuss the family of equations $\mathcal{O}$. Let us select an arbitrary member of this family, i.e. for some $s_0 \in Act^*$ and some $k \geq 0$, the equation

  $$x + s_0.x + \overline{s_0}^{(k)}(yes + no) = x + \overline{s_0}^{(k)}(yes + no) \ \ .$$

  We see that the depth of x is 1, the depth of $s_0.x$ is $|s_0| + 1$ and the depth of the term $\overline{s_0}^k(yes + no)$ is $(k+1)|s_0| + 1$ (which follows by the definition of the term $\overline{s}^k(m)$). We can also see that the term $\overline{s_0}^k(yes + no)$ accepts and rejects all traces of the form $s_0 s'$, where the length of s' is strictly bigger than $(k - 1)|s_0|$, which is enough for the statement to hold.

□

Now that we have defined the property we were looking for over a finite subset $\mathcal{E}$ of $\mathcal{E}'_{v,f}$, we proceed to show that the property itself is preserved by equational proofs from $\mathcal{E}$.

Theorem 9. *Let $\mathcal{E}$ be a finite subset of $\mathcal{E}'_{v,f}$ and let $m = n$ be an equation such that $\mathcal{E} \vdash m = n$. Assume that:*

- *$m \xrightarrow{s} m' + x$ for some string s, monitor m' and variable x and*
- *$n \not\xrightarrow{s} n' + x$ for any n'.*

Then, for every trace of the form $s.s'$ where $|s'| \geq depth(\mathcal{E})$, we have that $ss' \in L_a(\sigma(m))$ and $ss' \in L_r(\sigma(m))$ for every substitution σ.

Proof. We will use induction over the length of the proof that results in an arbitrary equation $m = n$. Our base case is a proof of length one, where the the only equations we can prove are the axioms themselves and therefore the property holds by Lemma 19.

Assume now we have shown that all proofs of length up to ℓ preserve the property. We will show that proofs of length up to $\ell + 1$ do so as well. The final step of a proof can be performed by applying:

- The congruence rule for $+$,
- The congruence rule for action prefixing $a._-$,
- A variable substitution (for an open substitution σ), or
- Transitivity.

Note here that, as we mentioned earlier, the axiom system $\mathcal{E}'_{v,f}$ is closed with respect to symmetry and therefore there is no need to use the symmetry rule in proofs. We proceed by considering each of the above-mentioned proof steps.

- The congruence rule for $+$ must be applied as so: Assume two equations $m_1 = n_1$ and $m_2 = n_2$, two already proven equations for which the statement of the theorem holds (inductive hypothesis). By applying the congruence rule for $+$ we have proven the equation $m = m_1 + m_2 = n_1 + n_2 = n$. Assume that $m \xrightarrow{s} m' + x$ for some string s, monitor m' and variable x and $n \not\xrightarrow{s} n' + x$ for any n'. By the operational semantics of Mon_F we have that either $m_1 \xrightarrow{s} m' + x$ or $m_2 \xrightarrow{s} m' + x$. Without loss of generality assume $m_1 \xrightarrow{s} m' + x$. Moreover we have that $n_1 \not\xrightarrow{s} n'_1 + x$ for any n'_1 since $n \not\xrightarrow{s} n' + x$ for any n'. By inductive hypothesis then for every trace of the form $s.s'$ where $|s'| \geq depth(\mathcal{E})$ we have that $s.s' \in L_a(\sigma(m_1))$ and $s.s' \in L_r(\sigma(m_1))$ for every substitution σ. This in turn implies that $s.s' \in L_a(\sigma(m))$ and $s.s' \in L_r(\sigma(m))$ for every substitution σ and we are done.

- We now consider the case of applying the congruence rule for action prefixing. Assume a proven equation $m_0 = n_0$ on which we apply the axiom prefixing congruence rule for an action $a \in Act$, that is, $m = a.m_0 = a.no = n$. Assume now that $m \xrightarrow{s} m_s + x$ for some string s, monitor m_s and variable x and $n \not\xrightarrow{s} n_s + x$ for any n_s. Since $m = a.m_0$, it follows that $s = as_0$ and $m_0 \xrightarrow{s_0} m'_0 + x$ for some m'_0 and $n_0 \not\xrightarrow{s_0} n'_0 + x$ for any

n'_0. Therefore by inductive hypothesis we have that all traces of the form $s_0.s'$ where $|s'| \geq \mathit{depth}(\mathcal{E})$ are accepted and rejected by m_0 under any substitution. Consequently all traces of the form $as_0.s' = ss'$ are both accepted and rejected by m under any substitution and we are done.

- Consider now variable substitution. Note that we will consider open substitutions, in order to capture the more general case. The case of closed substitutions is of course trivial as after one of them is applied there are no variable occurrences left in any equation and therefore the result holds vacuously. We have now that $\mathcal{E} \vdash m' = n'$ for some open monitors m' and n' and that we apply the open substitution σ_0 in order to prove the open equation $m = \sigma_0(m') = \sigma_0(n') = n$. Assume now that $\sigma_0(m) \xrightarrow{s} m_s + x$ for some string s, monitor m_s and variable x and $\sigma_0(n) \not\xrightarrow{s} n_s + x$ for any n_s. We can easily see that every such one-sided occurrence of a variable in the new equation must have resulted from a one-sided variable occurrence in $m' = n'$. This is because if there were no one-sided variable occurrences in the old equation, then under no substitution could one have introduced a variable in only one side without also introducing it on the other side. This means that there exists some variable y (which could be the same as x) such that $m' \xrightarrow{s_0} m'_{s_0} + y$ for some string s_0 where s_0 a prefix of s, monitor m'_{s_0} and variable y and $n' \not\xrightarrow{s_0} n'_{s_0} + y$ for any n'_{s_0}. The reason why s_0 must be a prefix of s is that an open substitution can only expand the traces that lead to a variable occurrence in the original term. By applying our inductive hypothesis on $m' = n'$, we have that both m' and n' must accept and reject all traces of the form $s_0.s'$ where $|s'| \geq \mathit{depth}(\mathcal{E})$ under any substitution σ. This, in turn, implies that $\sigma_0(m') = m$ accepts and rejects traces of the form ss' under any closed substitution σ. In fact $\sigma(\sigma_0(m')) = \sigma_0(\sigma_0(m'))$ which means that m and n reject the traces of the form $s_0.s$ as well. Since s_0 is a prefix of s we have that for every extension of s of length at least $\mathit{depth}(\mathcal{E})$ there exists an extension of s_0 of length at least $\mathit{depth}(\mathcal{E})$ that is a prefix of it. Since all traces $s_0.s'$ of this length are both accepted and rejected under any substitution, the same applies for the traces $s.s'$ and we are done.

- The case of transitivity is also straightforward though the following inductive argument. We start by $\mathcal{E} \vdash m = m'$ and $\mathcal{E} \vdash m' = n$ and we apply the transitivity rule to prove $m = n$. Assume that $m \xrightarrow{s} m_s + x$ for some trace s, variable x and monitor m_s, while $n \not\xrightarrow{s} n_s + x$ for any n_s. We have that either: $m' \xrightarrow{s} m'_s + x$ for some m'_s or $m' \not\xrightarrow{s} m'_s + x$. In the first case we have that the equation $m' = n$ which has already been proven by $\mathcal{E}$ satisfies the premises of the theorem and therefore by induction hypothesis all traces of the form $s.s'$ where $|s'| \geq \mathit{depth}(\mathcal{E})$ are both accepted and rejected by both m' and n. Since $n \simeq m$ by the soundness of $\mathcal{E}'_{v,f}$ and thus $\mathcal{E}$, we have that m also accept and rejects all of these traces and we are done. In the second case and via a similar argument we have the same result.

This concludes the case analysis for our inductive proof and we are done. $\square$

As we can see, if we start from any finite subset $\mathcal{E}$ of $\mathcal{E}'_{v,f}$, we are bound to only prove equations that have the property in the statement of Theorem 9. We now argue that for each $\mathcal{E}$ there will always exist sound equations in $\mathcal{E}'_{v,f}$ that do not satisfy the above property and therefore the axiom set $\mathcal{E}$ is not enough to prove them.

Lemma 20. *Let $\mathcal{E}$ be a finite subset of $\mathcal{E}'_{v,f}$. There exists a sound equation $m = n$ in $\mathcal{O}$ such that $m \xrightarrow{s} m' + x$ for some string s, monitor m' and variable x and $n \not\xrightarrow{s} n' + x$ for any n' and there is at least one trace of the form $s.s'$ where $|s'| \geq depth(\mathcal{E})$ and $s.s' \notin L_a(\sigma(m))$ and $s.s' \notin L_a(\sigma(m))$ for the one substitution $\sigma_{end} = end$, for every x.*

Proof. It suffices to give an example from the members of the family $\mathcal{O}$. Namely we consider the equation:

$$x + a^n.x + \overline{(a^n)}^3(yes + no) = x + \overline{(a^n)}^3(yes + no) \ ,$$

where $n > depth(\mathcal{E})$.

We can clearly see that first of all the occurrence of x after the trace a^n is one-sided in the left hand side of the equation. However there is a substitution (namely $\sigma(x) = end$) under which the trace a^{2n+1} is neither accepted nor rejected by the two monitors even though the length of $a^{(n+1)}$ is strictly larger than $depth(\mathcal{E})$. □

Theorem 10. *There is no finite complete set of axioms for verdict equivalence over Mon_F over a finite, non-unary set of actions.*

Proof. Let $\mathcal{E}$ be a finite subset $\mathcal{E}'_{v,f}$. Then, by the above lemma, $\mathcal{E}$ cannot prove the sound equation

$$x + a^n.x + \overline{(a^n)}^3(yes + no) = x + \overline{(a^n)}^3(yes + no) \ ,$$

for $n > depth(\mathcal{E})$ and we are done. □

6. Conclusions

In this article, we have studied the equational theory of recursion-free, regular monitors from [1, 2, 26] modulo two natural notions of monitor equivalence, namely verdict and ω-verdict equivalence. We have provided complete axiomatizations for those equivalences over closed and open terms. The axiomatizations over closed terms are finite when so is the set of actions monitors can process. On the other hand, even when the set of actions is finite, whether those equivalences have finite bases over open terms depends on the cardinality of the action set. For instance, we have shown that verdict equivalence has no finite equational axiomatization when the set of actions contains at least two actions.

Since verdict and ω-verdict equivalence are trace-based behavioral equivalences, our axiomatizations, which are summarized in Table 4, share a number of

equations with those for trace and completed trace equivalence over BCCSP [27] and for equality of regular expressions [20, 38, 49]. However, the presence of the *yes*, *no* and *end* verdicts yields a number of novelties and technical complications, which are most evident in the axiomatization results over open terms and in the negative result we present in Section 5. By way of example, we remark here that, as mentioned in [19], trace and completed trace equivalence are finitely based over BCCSP when the set of actions is finite, unlike the notions we study in this paper over monitors. Moreover, unlike the one given in this paper, proofs of non-finite-axiomatizability results for regular expressions rely on families of equations that exploit the interplay between Kleene star and concatenation, such as

$$a^* = (a^n)^*(1 + a + \cdots + a^{n-1}) \quad (n > 0).$$

See, for instance, [6, 20, 48].

The results presented in this article deal with a minimal language for monitors that is mainly of theoretical interest and set the stage for further research. An interesting and natural avenue for future work is to study the complexity of the equational theory of verdict and ω-verdict equivalence. Moreover, one could investigate axiomatizations of those behavioral equivalences over extensions of recursion-free monitors with the parallel operators considered in [1] and/or with recursion [26]. As shown in [1](Proposition 3.8), every 'reactive parallel monitor' is verdict equivalent to a regular one. This opens the tantalizing possibility that verdict equivalence affords an elegant equational axiomatization over such monitors. However, the proof of Proposition 3.8 in [1] relies on a non-trivial automata-theoretic construction, which would have to be simulated equationally to transform 'reactive parallel monitors' into regular ones. We leave this interesting problem for further study.

References

[1] Aceto, L., Achilleos, A., Francalanza, A., Ingólfsdóttir, A., & Lehtinen, K. (2019). Adventures in monitorability: from branching to linear time and back again. *Proc. ACM Program. Lang.*, *3*, 52:1–52:29. doi:10.1145/3290365.

[2] Aceto, L., Achilleos, A., Francalanza, A., Ingólfsdóttir, A., & Lehtinen, K. (2019). An operational guide to monitorability. In P. C. Ölveczky, & G. Salaün (Eds.), *Software Engineering and Formal Methods - 17th International Conference, SEFM 2019, Oslo, Norway, September 18-20, 2019, Proceedings* (pp. 433–453). Springer volume 11724 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-030-30446-1_23.

[3] Aceto, L., Attard, D. P., Francalanza, A., & Ingólfsdóttir, A. (2021). On benchmarking for concurrent runtime verification. In E. Guerra, & M. Stoelinga (Eds.), *Fundamental Approaches to Software Engineering - 24th International Conference, FASE 2021, Held as Part of the European*

Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings (pp. 3–23). Springer volume 12649 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-030-71500-7_1.

[4] Aceto, L., Castiglioni, V., Fokkink, W. J., Ingólfsdóttir, A., & Luttik, B. (2021). Are two binary operators necessary to finitely axiomatise parallel composition? In C. Baier, & J. Goubault-Larrecq (Eds.), *29th EACSL Annual Conference on Computer Science Logic, CSL 2021, January 25-28, 2021, Ljubljana, Slovenia (Virtual Conference)* (pp. 8:1–8:17). Schloss Dagstuhl - Leibniz-Zentrum für Informatik volume 183 of *LIPIcs*. doi:10.4230/LIPIcs.CSL.2021.8.

[5] Aceto, L., Castiglioni, V., Ingólfsdóttir, A., Luttik, B., & Pedersen, M. R. (2020). On the axiomatisability of parallel composition: A journey in the spectrum. In I. Konnov, & L. Kovács (Eds.), *31st International Conference on Concurrency Theory, CONCUR 2020* (pp. 18:1–18:22). Schloss Dagstuhl - Leibniz-Zentrum für Informatik volume 171 of *LIPIcs*. doi:10.4230/LIPIcs.CONCUR.2020.18.

[6] Aceto, L., Fokkink, W. J., & Ingólfsdóttir, A. (1998). On a question of A. Salomaa: The equational theory of regular expressions over a singleton alphabet is not finitely based. *Theoretical Computer Science*, *209*, 163–178. doi:10.1016/S0304-3975(97)00104-7.

[7] Aceto, L., Fokkink, W. J., Ingólfsdóttir, A., & Luttik, B. (2005). Finite equational bases in process algebra: Results and open questions. In A. Middeldorp, V. van Oostrom, F. van Raamsdonk, & R. C. de Vrijer (Eds.), *Processes, Terms and Cycles: Steps on the Road to Infinity, Essays Dedicated to Jan Willem Klop, on the Occasion of His 60th Birthday* (pp. 338–367). Springer volume 3838 of *Lecture Notes in Computer Science*. doi:10.1007/11601548_18.

[8] Baeten, J. C. M., Basten, T., & Reniers, M. A. (2009). *Process Algebra: Equational Theories of Communicating Processes* volume 50 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press. doi:10.1017/CBO9781139195003.

[9] Baeten, J. C. M., & Bergstra, J. A. (1990). Process algebra with a zero object. In J. C. M. Baeten, & J. W. Klop (Eds.), *CONCUR '90, Theories of Concurrency: Unification and Extension, Amsterdam, The Netherlands, August 27-30, 1990, Proceedings* (pp. 83–98). Springer volume 458 of *Lecture Notes in Computer Science*. doi:10.1007/BFb0039053.

[10] Barringer, H., Falcone, Y., Havelund, K., Reger, G., & Rydeheard, D. E. (2012). Quantified event automata: Towards expressive and efficient runtime monitors. In D. Giannakopoulou, & D. Méry (Eds.), *FM 2012: Formal Methods - 18th International Symposium* (pp. 68–84). Springer volume 7436

of *Lecture Notes in Computer Science*. doi:10.1007/978-3-642-32759-9_9.

[11] Barringer, H., Goldberg, A., Havelund, K., & Sen, K. (2004). Rule-based runtime verification. In B. Steffen, & G. Levi (Eds.), *Verification, Model Checking, and Abstract Interpretation, 5th International Conference, VMCAI 2004* (pp. 44–57). Springer volume 2937 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-540-24622-0_5.

[12] Barringer, H., Rydeheard, D. E., & Havelund, K. (2010). Rule systems for run-time monitoring: From Eagle to RuleR. *Journal of Logic and Computation*, *20*, 675–706. doi:10.1093/logcom/exn076.

[13] Bartocci, E., & Falcone, Y. (Eds.) (2018). *Lectures on Runtime Verification - Introductory and Advanced Topics* volume 10457 of *Lecture Notes in Computer Science*. Springer. doi:10.1007/978-3-319-75632-5.

[14] Bauer, A., Küster, J., & Vegliach, G. (2015). The ins and outs of first-order runtime verification. *Formal Methods in System Design*, *46*, 286–316. doi:10.1007/s10703-015-0227-2.

[15] Bauer, A., Leucker, M., & Schallhart, C. (2011). Runtime verification for LTL and TLTL. *ACM Trans. Softw. Eng. Methodol.*, *20*, 14:1–14:64. doi:10.1145/2000799.2000800.

[16] Bergstra, J. A., & Klop, J. W. (1984). Process algebra for synchronous communication. *Information and Control*, *60*, 109–137. doi:10.1016/S0019-9958(84)80025-X.

[17] Bonakdarpour, B., Fraigniaud, P., Rajsbaum, S., Rosenblueth, D. A., & Travers, C. (2016). Decentralized asynchronous crash-resilient runtime verification. In J. Desharnais, & R. Jagadeesan (Eds.), *27th International Conference on Concurrency Theory, CONCUR 2016* (pp. 16:1–16:15). Schloss Dagstuhl - Leibniz-Zentrum für Informatik volume 59 of *LIPIcs*. doi:10.4230/LIPIcs.CONCUR.2016.16.

[18] Brookes, S. D. (1983). A semantics and proof system for communicating processes. In E. M. Clarke, & D. Kozen (Eds.), *Logics of Programs* (pp. 68–85). Springer volume 164 of *Lecture Notes in Computer Science*. doi:10.1007/3-540-12896-4_356.

[19] Chen, T., Fokkink, W. J., Luttik, B., & Nain, S. (2008). On finite alphabets and infinite bases. *Information and Computation*, *206*, 492–519. doi:10.1016/j.ic.2007.09.003.

[20] Conway, J. H. (1971). *Regular Algebra and Finite Machines*. London: Chapman and Hall.

[21] Cranen, S., Groote, J. F., Keiren, J. J. A., Stappers, F. P. M., de Vink, E. P., Wesselink, W., & Willemse, T. A. C. (2013). An overview of the mCRL2 toolset and its recent advances. In N. Piterman, & S. A. Smolka (Eds.), *Tools and Algorithms for the Construction and Analysis of Systems - 19th International Conference, TACAS 2013* (pp. 199–213). Springer volume 7795 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-642-36742-7_15.

[22] Diekert, V., & Gastin, P. (2008). First-order definable languages. In J. Flum, E. Grädel, & T. Wilke (Eds.), *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]* (pp. 261–306). Amsterdam University Press volume 2 of *Texts in Logic and Games*.

[23] Falcone, Y., Fernandez, J., & Mounier, L. (2012). What can you verify and enforce at runtime? *International Journal on Software Tools for Technology Transfer*, *14*, 349–382. doi:10.1007/s10009-011-0196-8.

[24] Falcone, Y., Havelund, K., & Reger, G. (2013). A tutorial on runtime verification. In M. Broy, D. A. Peled, & G. Kalus (Eds.), *Engineering Dependable Software Systems* (pp. 141–175). IOS Press volume 34 of *NATO Science for Peace and Security Series, D: Information and Communication Security*. doi:10.3233/978-1-61499-207-3-141.

[25] Francalanza, A., Aceto, L., Achilleos, A., Attard, D. P., Cassar, I., Della Monica, D., & Ingólfsdóttir, A. (2017). A foundation for runtime monitoring. In S. Lahiri, & G. Reger (Eds.), *Runtime verification. RV* (pp. 8–29). Springer volume 10548 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-319-67531-2_2.

[26] Francalanza, A., Aceto, L., & Ingolfsdottir, A. (2017). Monitorability for the Hennessy–Milner Logic with recursion. *Formal Methods in System Design*, *51*, 87–116. doi:10.1007/s10703-017-0273-z.

[27] van Glabbeek, R. J. (2001). The linear time - branching time spectrum I. In *Handbook of Process Algebra* (pp. 3–99). North-Holland / Elsevier. doi:10.1016/b978-044482830-9/50019-9.

[28] Grabmayer, C., & Fokkink, W. (2020). A complete proof system for 1-free regular expressions modulo bisimilarity. In H. Hermanns, L. Zhang, N. Kobayashi, & D. Miller (Eds.), *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science* (pp. 465–478). ACM. doi:10.1145/3373718.3394744.

[29] Groote, J. F., & Reniers, M. A. (2001). Algebraic process verification. In J. A. Bergstra, A. Ponse, & S. A. Smolka (Eds.), *Handbook of Process Algebra* (pp. 1151–1208). North-Holland / Elsevier. doi:10.1016/b978-044482830-9/50035-7.

[30] Havelund, K., & Goldberg, A. (2005). Verify your runs. In B. Meyer, & J. Woodcock (Eds.), *Verified Software: Theories, Tools, Experiments, First IFIP TC, 2/WG 2.3 Conference, VSTTE 2005* (pp. 374–383). Springer volume 4171 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-540-69149-5_40.

[31] Havelund, K., & Rosu, G. (2001). Monitoring Java programs with Java PathExplorer. *Electron. Notes Theor. Comput. Sci.*, *55*, 200–217. doi:10.1016/S1571-0661(04)00253-1.

[32] Heering, J. (1986). Partial evaluation and ω-completeness of algebraic specifications. *Theor. Comput. Sci.*, *43*, 149–167. doi:10.1016/0304-3975(86)90173-8.

[33] Hennessy, M. (1981). A term model for synchronous processes. *Information and Control*, *51*, 58–75. doi:10.1016/S0019-9958(81)90082-6.

[34] Hennessy, M., & Milner, R. (1985). Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, *32*, 137–161. doi:10.1145/2455.2460.

[35] Hoare, C. A. R., Hayes, I. J., He, J., Morgan, C., Roscoe, A. W., Sanders, J. W., Sørensen, I. H., Spivey, J. M., & Sufrin, B. (1987). Laws of programming. *Communications of the ACM*, *30*, 672–686. doi:10.1145/27651.27653.

[36] Kamp, H. (1968). *Tense Logic and the Theory of Linear Order*. Ph.D. thesis UCLA.

[37] Kappé, T., Brunet, P., Silva, A., Wagemaker, J., & Zanasi, F. (2020). Concurrent kleene algebra with observations: From hypotheses to completeness. In J. Goubault-Larrecq, & B. König (Eds.), *Foundations of Software Science and Computation Structures - 23rd International Conference, FOSSACS 2020* (pp. 381–400). Springer volume 12077 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-030-45231-5_20.

[38] Kozen, D. (1994). A completeness theorem for Kleene algebras and the algebra of regular events. *Information and Computation*, *110*, 366–390. doi:10.1006/inco.1994.1037.

[39] Kozen, D., & Silva, A. (2020). Left-handed completeness. *Theoretical Computer Science*, *807*, 220–233. doi:10.1016/j.tcs.2019.10.040.

[40] Leucker, M., & Schallhart, C. (2009). A brief account of runtime verification. *Journal of Logical and Algebraic Methods in Programming*, *78*, 293–303. doi:10.1016/j.jlap.2008.08.004.

[41] Lin, H. (1995). PAM: A process algebra manipulator. *Formal Methods in System Design*, *7*, 243–259. doi:10.1007/BF01384078.

[42] Milner, R. (1980). *A Calculus of Communicating Systems* volume 92 of *Lecture Notes in Computer Science*. Springer. doi:10.1007/3-540-10235-3.

[43] Milner, R. (1984). A complete inference system for a class of regular behaviours. *Journal of Computer and System Sciences*, *28*, 439–466. doi:10.1016/0022-0000(84)90023-0.

[44] Milner, R. (1989). *Communication and Concurrency*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc. doi:10.5555/534666.

[45] Peled, D., & Havelund, K. (2018). Refining the safety-liveness classification of temporal properties according to monitorability. In T. Margaria, S. Graf, & K. G. Larsen (Eds.), *Models, Mindsets, Meta: The What, the How, and the Why Not? - Essays Dedicated to Bernhard Steffen on the Occasion of His 60th Birthday* (pp. 218–234). Springer volume 11200 of *Lecture Notes in Computer Science*. doi:10.1007/978-3-030-22348-9_14.

[46] Pnueli, A., & Zaks, A. (2006). PSL model checking and run-time verification via testers. In J. Misra, T. Nipkow, & E. Sekerinski (Eds.), *FM 2006: Formal Methods, 14th International Symposium on Formal Methods* (pp. 573–586). Springer volume 4085 of *Lecture Notes in Computer Science*. doi:10.1007/11813040_38.

[47] Reddy, S., Lemieux, C., Padhye, R., & Sen, K. (2020). Quickly generating diverse valid test inputs with reinforcement learning. In G. Rothermel, & D. Bae (Eds.), *ICSE '20: 42nd International Conference on Software Engineering, Seoul, South Korea, 27 June - 19 July, 2020* (pp. 1410–1421). ACM. doi:10.1145/3377811.3380399.

[48] Redko, V. (1964). On defining relations for the algebra of regular events. *Ukrainskiĭ matematicheskiĭ Zhurnal*, *16*, 120–126 (in Russian).

[49] Salomaa, A. (1966). Two complete axiom systems for the algebra of regular events. *Journal of the ACM*, *13*, 158–169. doi:10.1145/321312.321326.

[50] Schützenberger, M. P. (1965). On finite monoids having only trivial subgroups. *Inf. Control.*, *8*, 190–194. doi:10.1016/S0019-9958(65)90108-7.

[51] Sokolsky, O., & Rosu, G. (2012). Introduction to the special issue on runtime verification. *Formal Methods Syst. Des.*, *41*, 233–235. doi:10.1007/s10703-012-0174-0.

[52] Tabakov, D., Rozier, K. Y., & Vardi, M. Y. (2012). Optimized temporal monitors for SystemC. *Formal Methods Syst. Des.*, *41*, 236–268. doi:10.1007/s10703-011-0139-8.