Rule Formats for Determinism and Idempotency*

Luca Aceto[1], Arnar Birgisson[1], Anna Ingolfsdottir[1],
MohammadReza Mousavi[2], and Michel A. Reniers[2]

[1] School of Computer Science, Reykjavik University,
Kringlan 1, IS-103 Reykjavik, Iceland
[2] Department of Computer Science, Eindhoven University of Technology,
P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands

Abstract. Determinism is a semantic property of (a fragment of) a language that specifies that a program cannot evolve operationally in several different ways. Idempotency is a property of binary composition operators requiring that the composition of two identical specifications or programs will result in a piece of specification or program that is equivalent to the original components. In this paper, we propose two (related) meta-theorems for guaranteeing determinism and idempotency of binary operators. These meta-theorems are formulated in terms of syntactic templates for operational semantics, called rule formats. We show the applicability of our formats by applying them to various operational semantics from the literature.

1 Introduction

Structural Operational Semantics (SOS) [18] is a popular method for assigning a rigorous meaning to specification and programming languages. The meta-theory of SOS provides powerful tools for proving semantic properties for such languages without investing too much time on the actual proofs; it offers syntactic templates for SOS rules, called *rule formats*, which guarantee semantic properties once the SOS rules conform to the templates (see, e.g., the references [1, 16] for surveys on the meta-theory of SOS). There are various rule formats in the literature for many different semantic properties, ranging from basic properties such as commutativity [14] and associativity [6] of operators, and congruence of behavioral equivalences (see, e.g., [22]) to more technical and involved ones such as non-interference [19] and (semi-)stochasticity [12]. In this paper, we propose rule formats for two (related) properties, namely, determinism and idempotency.

Determinism is a semantic property of (a fragment of) a language that specifies that a program cannot evolve operationally in several different ways. It holds

* The work of Aceto, Birgisson and Ingolfsdottir has been partially supported by the projects "The Equational Logic of Parallel Processes" (nr. 060013021), and "New Developments in Operational Semantics" (nr. 080039021) of the Icelandic Research Fund. Birgisson has been further supported by a research-student grant nr. 080890008 of the Icelandic Research Fund.

for sub-languages of many process calculi and programming languages, and it is also a crucial property for many formalisms for the description of timed systems, where time transitions are required to be deterministic, because the passage of time should not resolve any choice.

Idempotency is a property of binary composition operators requiring that the composition of two identical specifications or programs will result in a piece of specification or program that is equivalent to the original components. Idempotency of a binary operator f is concisely expressed by the following algebraic equation.

$$f(x, x) = x$$

Determinism and idempotency may seem unrelated at first sight. However, it turns out that in order to obtain a powerful rule format for idempotency, we need to have the determinism of certain transition relations in place. Therefore, having a syntactic condition for determinism, apart from its intrinsic value, results in a powerful, yet syntactic framework for idempotency.

To our knowledge, our rule format for idempotency has no precursor in the literature. As for determinism, in [8], a rule format for bounded nondeterminism is presented but the case for determinism is not studied. Also, in [20] a rule format is proposed to guarantee several time-related properties, including time determinism, in the settings of Ordered SOS. In case of time determinism, their format corresponds to a subset of our rule format when translated to the setting of ordinary SOS, by means of the recipe given in [13].

We made a survey of existing deterministic process calculi and of idempotent binary operators in the literature and we have applied our formats to them. Our formats could cover all practical cases that we have discovered so far, which is an indication of its expressiveness and relevance.

The rest of this paper is organized as follows. In Section 2 we recall some basic definitions from the meta-theory of SOS. In Section 3, we present our rule format for determinism and prove that it does guarantee determinism for certain transition relations. Section 4 introduces a rule format for idempotency and proves it correct. In Sections 3 and 4, we also provide several examples to motivate the constraints of our rule formats and to demonstrate their practical applications. Finally, Section 5 concludes the paper and presents some directions for future research.

2 Preliminaries

In this section we present, for sake of completeness, some standard definitions from the meta-theory of SOS that will be used in the remainder of the paper.

Definition 1 (Signature and terms). *We let V represent an infinite set of variables and use $x, x', x_i, y, y', y_i, \ldots$ to range over elements of V. A* signature *Σ is a set of function symbols, each with a fixed arity. We call these symbols* operators *and usually represent them by $f, g, \ldots$. An operator with arity zero is called a* constant. *We define the set $\mathbb{T}(\Sigma)$ of* terms *over Σ as the smallest set satisfying the following constraints.*

- *A variable $x \in V$ is a term.*
- *If $f \in \Sigma$ has arity n and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.*

We use $t, t', t_i, \ldots$ to range over terms. We write $t_1 \equiv t_2$ if t_1 and t_2 are syntactically equal. The function $vars : \mathbb{T}(\Sigma) \to 2^V$ gives the set of variables appearing in a term. The set $\mathbb{C}(\Sigma) \subseteq \mathbb{T}(\Sigma)$ is the set of closed terms*, i.e., terms that contain no variables. We use $p, p', p_i, \ldots$ to range over closed terms. A* substitution *σ is a function of type $V \to \mathbb{T}(\Sigma)$. We extend the domain of substitutions to terms homomorphically. If the range of a substitution lies in $\mathbb{C}(\Sigma)$, we say that it is a* closing substitution*.*

Definition 2 (Transition System Specifications (TSS), formulae and transition relations). *A* transition system specification *is a triplet (Σ, L, D) where*

- *Σ is a signature.*
- *L is a set of labels. If $l \in L$, and $t, t' \in \mathbb{T}(\Sigma)$ we say that $t \xrightarrow{l} t'$ is a* positive formula *and $t \overset{l}{\nrightarrow}$ is a* negative formula*. A formula, typically denoted by ϕ, ψ, ϕ', ϕ_i, ... is either a negative formula or a positive one.*
- *D is a set of* deduction rules*, i.e., tuples of the form (Φ, ϕ) where Φ is a set of formulae and ϕ is a positive formula. We call the formulae contained in Φ the* premises *of the rule and ϕ the* conclusion*.*

We write $vars(r)$ to denote the set of variables appearing in a deduction rule (r)*. We say a formula is* closed *if all of its terms are closed. Substitutions are also extended to formulae and sets of formulae in the natural way. A set of positive closed formulae is called a* transition relation*.*

We often refer to a formula $t \xrightarrow{l} t'$ as a *transition* with t being its *source*, l its label, and t' its *target*. A deduction rule (Φ, ϕ) is typically written as $\frac{\Phi}{\phi}$. For a deduction rule r, we write $conc(r)$ to denote its conclusion and $prem(r)$ to denote its premises. We call a deduction rule *f-defining* when the outermost function symbol appearing in its source of the conclusion is f.

The meaning of a TSS is defined by the following notion of least three-valued stable model. To define this notion, we need two auxiliary definitions, namely provable transition rules and contradiction, which are given below.

Definition 3 (Provable Transition Rules). *A deduction rule is called a* transition rule *when it is of the form $\frac{N}{\phi}$ with N a set of* negative formulae*. A TSS $\mathcal{T}$* proves *$\frac{N}{\phi}$, denoted by $\mathcal{T} \vdash \frac{N}{\phi}$, when there is a well-founded upwardly branching tree with formulae as nodes and of which*

- *the root is labelled by ϕ;*
- *if a node is labelled by ψ and the nodes above it form the set K then:*
  - *ψ is a negative formula and $\psi \in N$, or*
  - *ψ is a positive formula and $\frac{K}{\psi}$ is an instance of a deduction rule in $\mathcal{T}$.*

Definition 4 (Contradiction and Contingency). *Formula $t \xrightarrow{l} t'$ is said to* contradict *$t \overset{l}{\nrightarrow}$, and vice versa. For two sets Φ and Ψ of formulae, Φ* contradicts

Ψ, denoted by $\Phi \not\models \Psi$, when there is a $\phi \in \Phi$ that contradicts a $\psi \in \Psi$. Φ is contingent *w.r.t. Ψ, denoted by $\Phi \models \Psi$, when Φ does not contradict Ψ.*

It immediately follows from the above definition that contradiction and contingency are symmetric relations on (sets of) formulae. We now have all the necessary ingredients to define the semantics of TSSs in terms of three-valued stable models.

Definition 5 (The Least Three-Valued Stable Model). *A pair (C, U) of sets of positive closed transition formulae is called a* three-valued stable model *for a TSS $\mathcal{T}$ when*

- *for all $\phi \in C$, $\mathcal{T} \vdash \frac{N}{\phi}$ for a set N such that $C \cup U \models N$, and*
- *for all $\phi \in U$, $\mathcal{T} \vdash \frac{N}{\phi}$ for a set N such that $C \models N$.*

C stands for Certainly *and U for* Unknown*; the third value is determined by the formulae not in $C \cup U$. The* least *three-valued stable model is a three valued stable model which is the least with respect to the ordering on pairs of sets of formulae defined as $(C, U) \leq (C', U')$ iff $C \subseteq C'$ and $U' \subseteq U$. When for the least three-valued stable model it holds that $U = \emptyset$, we say that $\mathcal{T}$ is* complete*.*

Complete TSSs univocally define a transition relation, i.e., the C component of their least three-valued stable model. Completeness is central to almost all meta-results in the SOS meta-theory and, as it turns out, it also plays an essential role in our meta-results concerning determinism and idempotency. All practical instances of TSSs are complete and there are syntactic sufficient conditions guaranteeing completeness, see for example [9].

3 Determinism

Definition 6 (Determinism). *A transition relation T is called deterministic for label l, when if $p \xrightarrow{l} p' \in T$ and $p \xrightarrow{l} p'' \in T$, then $p' \equiv p''$.*

Before we define a format for determinism, we need two auxiliary definitions. The first one is the definition of source dependent variables, which we borrow from [15] with minor additions.

Definition 7 (Source dependency). *For a deduction rule, we define the set of* source dependent *variables as the smallest set that contains*

1. *all variables appearing in the source of the conclusion, and*
2. *all variables that appear in the target of a premise where all variables in the source of that premise are source dependent.*

For a source dependent variable v, let $\mathcal{R}$ be the collection of transition relations appearing in a set of premises needed to show source dependency through condition 2. We say that v is source dependent via *the relations in $\mathcal{R}$.*

Note that for a source dependent variable, the set $\mathcal{R}$ is not necessarily unique. For example, in the rule

$$\frac{y \xrightarrow{l_1} y' \quad x \xrightarrow{l_2} z \quad z \xrightarrow{l_3} y'}{f(x,y) \xrightarrow{l} y'}$$

the variable y' is source dependent both via the set $\{\xrightarrow{l_1}\}$ as well as $\{\xrightarrow{l_2}, \xrightarrow{l_3}\}$.

The second auxiliary definition needed for our determinism format is the definition of determinism-respecting substitutions.

Definition 8 (Determinism-Respecting Pairs of Substitutions). *Given a set L of labels, a pair of substitutions (σ, σ') is determinism-respecting w.r.t. a pair of sets of formulae (Φ, Φ') and L when for all two positive formulae $s \xrightarrow{l} s' \in \Phi$ and $t \xrightarrow{l} t' \in \Phi'$ such that $l \in L$, $\sigma(s) \equiv \sigma'(t)$ only if $\sigma(s') \equiv \sigma'(t')$.*

Definition 9 (Determinism Format). *A TSS $\mathcal{T}$ is in the determinism format w.r.t. a set of labels L, when for each $l \in L$ the following conditions hold.*

1. *In each deduction rule $\frac{\Phi}{t \xrightarrow{l} t'}$, each variable $v \in vars(t')$ is source dependent via a subset of $\{\xrightarrow{l} \mid l \in L\}$, and*
2. *for each pair of distinct deduction rules $\frac{\Phi_0}{t_0 \xrightarrow{l} t_0'}$ and $\frac{\Phi_1}{t_1 \xrightarrow{l} t_1'}$ and for each determinism-respecting pair of substitutions (σ, σ') w.r.t. (Φ_0, Φ_1) and L such that $\sigma(t_0) \equiv \sigma'(t_1)$, it holds that either $\sigma(t_0') \equiv \sigma'(t_1')$ or $\sigma(\Phi_0)$ contradicts $\sigma'(\Phi_1)$.*

The first constraint in the definition above ensures that each rule in a TSS in the determinism format, with some $l \in L$ as its label of conclusion, can be used to prove at most one outgoing transition for each closed term. The second requirement guarantees that no two different rules can be used to prove two distinct l-labelled transitions for any closed term.

Theorem 1. *Consider a TSS with (C, U) as its least three-valued stable model and a subset L of its labels. If the TSS is in the determinism format w.r.t. L, then C is deterministic for each $l \in L$.*

For a TSS in the determinism format with (C, U) as its least three-valued stable model, U and thus $C \cup U$ need not be deterministic. The following counter-example illustrates this phenomenon.

Example 1. Consider the TSS given by the following deduction rules.

$$\frac{a \xrightarrow{l} a}{a \xrightarrow{l} b} \qquad \frac{a \xrightarrow{l}}{a \xrightarrow{l} a}$$

The above-given TSS is in the determinism format since $a \xrightarrow{l} a$ and $a \xrightarrow{l}$ contradict each other (under any substitution). Its least three-valued stable model is, however, $(\emptyset, \{a \xrightarrow{l} a, a \xrightarrow{l} b\})$ and $\{a \xrightarrow{l} a, a \xrightarrow{l} b\}$ is not deterministic.

Corollary 1. *Consider a complete TSS with L as a subset of its labels. If the TSS is in the determinism format w.r.t. L, then its defined transition relation is deterministic for each $l \in L$.*

Constraint 2 in Definition 9 may seem difficult to verify, since it requires checks for all possible (determinism-respecting) substitutions. However, in practical cases, to be quoted in the remainder of this paper, variable names are chosen in such a way that constraint 2 can be checked syntactically. For example, consider the following two deduction rules.

$$\frac{x \xrightarrow{a} x'}{f(x,y) \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} \quad x \xrightarrow{b} x'}{f(y,x) \xrightarrow{a} x'}$$

If in both deduction rules $f(x,y)$ (or symmetrically $f(y,x)$) was used, it could have been easily seen from the syntax of the rules that the premises of one deduction rule always (under all pairs of substitutions agreeing on the value of x) contradict the premises of the other deduction rule and, hence, constraint 2 is trivially satisfied. Based on this observation, we next present a rule format, whose constraints have a purely syntactic form, and that is sufficiently powerful to handle all the examples we discuss in Section 3.1. (Note that, for the examples in Section 3.1, checking the constraints of Definition 9 is not too hard either.)

Definition 10 (Normalized TSSs). *A TSS is normalized w.r.t. L if each deduction rule is f-defining for some function symbol f, and for each label $l \in L$, each function symbol f and each pair of deduction rules of the form*

$$(\mathrm{r})\frac{\Phi_r}{f(\overrightarrow{s}) \xrightarrow{l} s'} \quad (\mathrm{r}')\frac{\Phi_{r'}}{f(\overrightarrow{t}) \xrightarrow{l} t'}$$

the following constraints are satisfied:

1. *the sources of the conclusions coincide, i.e., $f(\overrightarrow{s}) \equiv f(\overrightarrow{t})$,*
2. *each variable $v \in vars(s')$ (symmetrically $v \in vars(t')$) is source dependent in* (r) *(respectively in* (r′)*) via some subset of $\{\xrightarrow{l} \mid l \in L\}$,*
3. *for each variable $v \in vars(r) \cap vars(r')$ there is a set of formulae in $\Phi_r \cap \Phi_{r'}$ proving its source dependency (both in* (r) *and* (r′)*) via some subset of $\{\xrightarrow{l} \mid l \in L\}$.*

The second and third constraint in Definition 11 guarantee that the syntactic equivalence of relevant terms (the target of the conclusion or the premises negating each other) will lead to syntactically equivalent closed terms under all determinism-respecting pairs of substitutions.

The reader can check that all the examples quoted from the literature in Section 3.1 are indeed normalized TSSs.

Definition 11 (Syntactic Determinism Format). *A normalized TSS is in the (syntactic) determinism format w.r.t. L, when for each two deduction rules $\frac{\Phi_0}{f(\overrightarrow{s}) \xrightarrow{l} s'}$ and $\frac{\Phi_1}{f(\overrightarrow{s}) \xrightarrow{l} s''}$, it holds that $s' \equiv s''$ or Φ_0 contradicts Φ_1.*

The following theorem states that for normalized TSSs, Definition 11 implies Definition 9.

Theorem 2. *Each normalized TSS in the syntactic determinism format w.r.t. L is also in the determinism format w.r.t. L.*

For brevity, we omit the proof of Theorem 2. The following statement is thus a corollary to Theorems 2 and 1.

Corollary 2. *Consider a normalized TSS with (C, U) as its least three-valued stable model and a subset L of its labels. If the TSS is in the (syntactic) determinism format w.r.t. L (according to Definition 11), then C is deterministic w.r.t. any $l \in L$.*

3.1 Examples

In this section, we present some examples of various TSSs from the literature and apply our (syntactic) determinism format to them. Some of the examples we discuss below are based on TSSs with predicates. The extension of our formats with predicates is straightforward and we discuss it in Section 4.3 to follow.

Example 2 (Conjunctive Nondeterministic Processes). Hennessy and Plotkin, in [10], define a language, called conjunctive nondeterministic processes, for studying logical characterizations of processes. The signature of the language consists of a constant 0, a unary action prefixing operator $a.\_$ for each $a \in A$, and a binary conjunctive nondeterminism operator $\vee$. The operational semantics of this language is defined by the following deduction rules.

$$\frac{}{0 \ \mathrm{can}_a} \qquad \frac{}{a.x \ \mathrm{can}_a} \qquad \frac{x \ \mathrm{can}_a}{x \vee y \ \mathrm{can}_a} \qquad \frac{y \ \mathrm{can}_a}{x \vee y \ \mathrm{can}_a}$$

$$\frac{}{0 \ \mathrm{after}_a \ 0} \qquad \frac{}{a.x \ \mathrm{after}_a \ x} \qquad \frac{}{a.x \ \mathrm{after}_b \ 0} \ a \neq b \qquad \frac{x \ \mathrm{after}_a \ x' \quad y \ \mathrm{after}_a \ y'}{x \vee y \ \mathrm{after}_a \ x' \vee y'}$$

The above TSS is in the (syntactic) determinism format with respect to the transition relation after_a (for each $a \in A$). Hence, we can conclude that the transition relations after_a are deterministic.

Example 3 (Delayed choice). The second example we discuss is a subset of the process algebra $\mathrm{BPA}_{\delta\epsilon} + \mathrm{DC}$ [4], i.e., Basic Process Algebra with deadlock and empty process extended with delayed choice. First we restrict attention to the fragment of this process algebra without non-deterministic choice $+$ and with action prefix $a.\_$ instead of general sequential composition $\cdot$. This altered process algebra has the following deduction rules, where a ranges over the set of actions A:

$$\frac{}{\epsilon \downarrow} \qquad \frac{}{a.x \xrightarrow{a} x} \qquad \frac{x \downarrow}{x \mp y \downarrow} \qquad \frac{y \downarrow}{x \mp y \downarrow}$$

$$\frac{x \xrightarrow{a} x' \qquad y \xrightarrow{a} y'}{x \mp y \xrightarrow{a} x' \mp y'} \qquad \frac{x \xrightarrow{a} x' \qquad y \overset{a}{\not\to}}{x \mp y \xrightarrow{a} x'} \qquad \frac{x \overset{a}{\not\to} \qquad y \xrightarrow{a} y'}{x \mp y \xrightarrow{a} y'}$$

In the above specification, predicate $p \downarrow$ denotes the possibility of termination for process p. The intuition behind the delayed choice operator, denoted by $\_ \mp \_$, is that the choice between two components is only resolved when one performs an action that the other cannot perform. When both components can perform an action, the delayed choice between them remains unresolved and the two components synchronize on the common action. This transition system specification is in the (syntactic) determinism format w.r.t. $\{a \mid a \in A\}$.

Addition of non-deterministic choice $+$ or sequential composition $\cdot$ results in deduction rules that do not satisfy the determinism format. For example, addition of sequential composition comes with the following deduction rules:

$$\frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y} \qquad \frac{x \downarrow \quad y \xrightarrow{a} y'}{x \cdot y \xrightarrow{a} y'}$$

The sets of premises of these rules do not contradict each other. The extended TSS is indeed non-deterministic since, for example, $(\epsilon \mp (a.\epsilon)) \cdot (a.\epsilon) \xrightarrow{a} \epsilon$ and $(\epsilon \mp (a.\epsilon)) \cdot (a.\epsilon) \xrightarrow{a} \epsilon \cdot (a.\epsilon)$.

Example 4 (Time transitions I). This example deals with the Algebra of Timed Processes, ATP, of Nicollin and Sifakis [17]. In the TSS given below, we specify the time transitions (denoted by label χ) of delayable deadlock δ, non-deterministic choice $\_ \oplus \_$, unit-delay operator $\lfloor \_ \rfloor \_$ and parallel composition $\_ \parallel \_$.

$$\frac{}{\delta \xrightarrow{\chi} \delta} \qquad \frac{x \xrightarrow{\chi} x' \quad y \xrightarrow{\chi} y'}{x \oplus y \xrightarrow{\chi} x' \oplus y'} \qquad \frac{}{\lfloor x \rfloor(y) \xrightarrow{\chi} y} \qquad \frac{x \xrightarrow{\chi} x' \quad y \xrightarrow{\chi} y'}{x \parallel y \xrightarrow{\chi} x' \parallel y'}$$

These deduction rules all trivially satisfy the determinism format for time transitions since the sources of conclusions of different deduction rules cannot be unified. Also the additional operators involving time, namely, the delay operator $\lfloor \_ \rfloor^d \_$, execution delay operator $\lceil \_ \rceil^d \_$ and unbounded start delay operator $\lfloor \_ \rfloor^\omega$, satisfy the determinism format for time transitions. The deduction rules are given below, for $d \geq 1$:

$$\frac{}{\lfloor x \rfloor^1(y) \xrightarrow{\chi} y} \qquad \frac{x \xrightarrow{\chi} x'}{\lfloor x \rfloor^{d+1}(y) \xrightarrow{\chi} \lfloor x' \rfloor^d(y)} \qquad \frac{x \overset{\chi}{\not\to}}{\lfloor x \rfloor^{d+1}(y) \xrightarrow{\chi} \lfloor x \rfloor^d(y)}$$

$$\frac{x \xrightarrow{\chi} x'}{\lfloor x \rfloor^\omega \xrightarrow{\chi} \lfloor x' \rfloor^\omega} \qquad \frac{x \overset{\chi}{\not\to}}{\lfloor x \rfloor^\omega \xrightarrow{\chi} \lfloor x \rfloor^\omega}$$

$$\frac{x \xrightarrow{\chi} x'}{\lceil x \rceil^1(y) \xrightarrow{\chi} y} \qquad \frac{x \xrightarrow{\chi} x'}{\lceil x \rceil^{d+1}(y) \xrightarrow{\chi} \lceil x' \rceil^d(y)}$$

Example 5 (Time transitions II). Most of the timed process algebras that originate from the Algebra of Communicating Processes (ACP) from [5, 3] such as those reported in [2] have a deterministic time transition relation as well.

In the TSS given below, the time unit delay operator is denoted by $\sigma_{\mathrm{rel}}\_$, nondeterministic choice is denoted by $\_ + \_$, and sequential composition is denoted by $\_ \cdot \_$. The deduction rules for the time transition relation for this process algebra are the following:

$$\frac{}{\sigma_{\mathrm{rel}}(x) \xrightarrow{1} x} \qquad \frac{x \xrightarrow{1} x' \quad y \xrightarrow{1} y'}{x + y \xrightarrow{1} x' + y'} \qquad \frac{x \xrightarrow{1} x' \quad y \overset{1}{\not\to}}{x + y \xrightarrow{1} x'} \qquad \frac{x \overset{1}{\not\to} \quad y \xrightarrow{1} y'}{x + y \xrightarrow{1} y'}$$

$$\frac{x \xrightarrow{1} x' \quad x \not\downarrow}{x \cdot y \xrightarrow{1} x' \cdot y} \qquad \frac{x \xrightarrow{1} x' \quad y \overset{1}{\not\to}}{x \cdot y \xrightarrow{1} x' \cdot y} \qquad \frac{x \xrightarrow{1} x' \quad x \downarrow \quad y \xrightarrow{1} y'}{x \cdot y \xrightarrow{1} x' \cdot y + y'} \qquad \frac{x \overset{1}{\not\to} \quad x \downarrow \quad y \xrightarrow{1} y'}{x \cdot y \xrightarrow{1} y'}$$

Note that here we have an example of deduction rules, the first two deduction rules for time transitions of a sequential composition, for which the premises do not contradict each other. Still these deduction rules satisfy the determinism format since the targets of the conclusions are identical. In the syntactically richer framework of [21], where arbitrary first-order logic formulae over transitions are allowed, those two deduction rules are presented by a single rule with premise $x \xrightarrow{1} x' \wedge (x \not\downarrow \vee y \overset{1}{\not\to})$.

Sometimes such timed process algebras have an operator for specifying an arbitrary delay, denoted by $\sigma^*_{\mathrm{rel}}\_$, with the following deduction rules.

$$\frac{x \overset{1}{\not\to}}{\sigma^*_{\mathrm{rel}}(x) \xrightarrow{1} \sigma^*_{\mathrm{rel}}(x)} \qquad \frac{x \xrightarrow{1} x'}{\sigma^*_{\mathrm{rel}}(x) \xrightarrow{1} x' + \sigma^*_{\mathrm{rel}}(x)}$$

The premises of these rules contradict each other and so, the semantics of this operator also satisfies the constraints of our (syntactic) determinism format.

4 Idempotency

Our order of business in this section is to present a rule format that guarantees the idempotency of certain binary operators. In the definition of our rule format, we rely implicitly on the work presented in the previous section.

4.1 Format

Definition 12 (Idempotency). *A binary operator $f \in \Sigma$ is* idempotent w.r.t. an equivalence $\sim$ *on closed terms if and only if for each $p \in \mathbb{C}(\Sigma)$, it holds that $f(p, p) \sim p$.*

Idempotency is defined with respect to a notion of behavioral equivalence. There are various notions of behavioral equivalence defined in the literature, which are by and large, weaker than bisimilarity defined below. Thus, to be as general as possible, we prove our idempotency result for all notions that contain, i.e., are weaker than, bisimilarity.

Definition 13 (Bisimulation). *Let $\mathcal{T}$ be a TSS with signature Σ. A relation $\mathcal{R} \subseteq \mathbb{C}(\Sigma) \times \mathbb{C}(\Sigma)$ is a* bisimulation relation *if and only if $\mathcal{R}$ is symmetric and for all $p_0, p_1, p_0' \in \mathbb{C}(\Sigma)$ and $l \in L$*

$$(p_0 \,\mathcal{R}\, p_1 \wedge \mathcal{T} \vdash p_0 \xrightarrow{l} p_0') \Rightarrow \exists_{p_1' \in \mathbb{C}(\Sigma)} (\mathcal{T} \vdash p_1 \xrightarrow{l} p_1' \wedge p_0' \,\mathcal{R}\, p_1').$$

Two terms $p_0, p_1 \in \mathbb{C}(\Sigma)$ are called bisimilar*, denoted by $p_0 \leftrightarroweq p_1$, when there exists a bisimulation relation $\mathcal{R}$ such that $p_0 \mathcal{R} p_1$.*

Definition 14 (The Idempotency Rule Format). *Let $\gamma : L \times L \to L$ be a partial function such that $\gamma(l_1, l_2) \in \{l_1, l_2\}$ if it is defined. We define the following two rule forms.*

1_l. *Choice rules*

$$\frac{\{x_i \xrightarrow{l} t\} \cup \Phi}{f(x_0, x_1) \xrightarrow{l} t}, \quad i \in \{0, 1\}$$

$2_{l_0,l_1}$. *Communication rules*

$$\frac{\{x_0 \xrightarrow{l_0} t_0,\, x_1 \xrightarrow{l_1} t_1\} \cup \Phi}{f(x_0, x_1) \overset{\gamma(l_0,l_1)}{\to} f(t_0, t_1)}, \quad t_0 \equiv t_1 \text{ or } (l_0 = l_1 \text{ and } \xrightarrow{l_0} \text{ is deterministic })$$

In each case, Φ can be an arbitrary, possibly empty set of (positive or negative) formulae.

In addition, we define the starred version of each form, 1_l^* and $2_{l_0,l_1}^*$. The starred version of each rule is the same as the unstarred one except that t, t_0 and t_1 are restricted to be concrete variables and the set Φ must be empty in each case.

A TSS is in idempotency format w.r.t. a binary operator *f if each f-defining rule, i.e., a deduction rule with f appearing in the source of the conclusion, is of the forms 1_l or $2_{l_0,l_1}$, for some $l, l_0, l_1 \in L$, and for each label $l \in L$ there exists at least one rule of the forms 1_l^* or $2_{l,l}^*$.*

We should note that the starred versions of the forms are special cases of their unstarred counterparts; for example a rule which has form 1_l^* also has form 1_l.

Theorem 3. *Assume that a TSS is complete and is in the idempotency format with respect to a binary operator f. Then, f is idempotent w.r.t. to any equivalence $\sim$ such that $\leftrightarroweq \subseteq \sim$.*

4.2 Relaxing the restrictions

In this section we consider the constraints of the idempotency rule format and show that they cannot be dropped without jeopardizing the meta-theorem.

First of all we note that, in rule form 1_l, it is necessary that the label of the premise matches the label of the conclusion. If it does not, in general, we cannot prove that $f(p,p)$ simulates p or vice versa. This requirement can be stated more generally for both rule forms in Definition 14; the label of the conclusion must be among the labels of the premises. The requirement that $\gamma(l,l') \in \{l,l'\}$ exists to ensure this constraint for form $2_{l,l'}$. A simple synchronization rule provides a counter-example that shows why this restriction is needed. Consider the following TSS with constants 0, τ, a and $\bar{a}$ and two binary operators $+$ and $\|$.

$$\frac{}{\alpha \xrightarrow{\alpha} 0} \quad \frac{x \xrightarrow{\alpha} x'}{x+y \xrightarrow{\alpha} x'} \quad \frac{y \xrightarrow{\alpha} y'}{x+y \xrightarrow{\alpha} y'} \quad \frac{x \xrightarrow{a} x' \quad y \xrightarrow{\bar{a}} y'}{x \parallel y \xrightarrow{\tau} x' \parallel y'}$$

where α is τ, a or $\bar{a}$. Here it is easy to see that although $(a+\bar{a}) \parallel (a+\bar{a})$ has an outgoing τ-transition, $a+\bar{a}$ does not afford such a transition.

The condition that for each l at least one rule of the forms 1^*_l or $2^*_{l,l}$ must exist comprises a few constraints on the rule format. First of all, it says there must be at least one f-defining rule. If not, it is easy to see that there could exist a process p where $f(p,p)$ deadlocks (since there are no f-defining rules) but p does not. It also states that there must be at least one rule in the starred form, where the targets are restricted to variables. To motivate these constraints, consider the following TSS.

$$\frac{}{a \xrightarrow{a} 0} \quad \frac{x \xrightarrow{a} a}{f(x,y) \xrightarrow{a} a}$$

The processes a and $f(a,a)$ are not bisimilar as the former can do an a-transition but the latter is stuck. The starred forms also require that Φ is empty, i.e. there is no testing. This is necessary in the proof because in the presence of extra premises, we cannot in general instantiate such a rule to show that $f(p,p)$ simulates p. Finally, the condition requires that if we rely on a rule of the form $2^*_{l,l'}$ and $t_0 \not\equiv t_1$, then the labels l and l' in the premises of the rule must coincide. To see why, consider a TSS containing a *left synchronize* operator U, one that synchronizes a step from each operand but uses the label of the left one. Here we let $\alpha \in \{a, \bar{a}\}$.

$$\frac{}{\alpha \xrightarrow{\alpha} 0} \quad \frac{x \xrightarrow{\alpha} x'}{x+y \xrightarrow{\alpha} x'} \quad \frac{y \xrightarrow{\alpha} y'}{x+y \xrightarrow{\alpha} y'} \quad \frac{x \xrightarrow{a} x' \quad y \xrightarrow{\bar{a}} y'}{x \mathsf{U} y \xrightarrow{a} x' \mathsf{U} y'}$$

In this TSS the processes $(a+\bar{a})$ and $(a+\bar{a})\mathsf{U}\,(a+\bar{a})$ are not bisimilar since the latter does not afford an $\bar{a}$-transition whereas the former does.

For rules of form $2_{l,l'}$ we require that either $t_0 \equiv t_1$, or that the mentioned labels are the same and the associated transition relation is deterministic. This requirement is necessary in the proof to ensure that the target of the conclusion fits our definition of $\simeq_f$, i.e. the operator is applied to two identical terms. Consider the following TSS where $\alpha \in \{a, b\}$.

$$\frac{}{a \xrightarrow{a} a} \quad \frac{}{a \xrightarrow{a} b} \quad \frac{}{b \xrightarrow{b} b} \quad \frac{x \xrightarrow{\alpha} x' \quad y \xrightarrow{\alpha} y'}{x|y \xrightarrow{\alpha} x'|y'}$$

For the operator $|$, this violates the condition $t_0 \equiv t_1$ (note that $\xrightarrow{a}$ is not deterministic). We observe that $a|a \xrightarrow{a} a|b$. The only possibilities for a to simulate this a-transition is either with $a \xrightarrow{a} a$ or with $a \xrightarrow{a} b$. However, neither a nor b can be bisimilar to $a|b$ because both a and b have outgoing transitions while $a|b$ is stuck. Therefore a and $a|a$ cannot be bisimilar. If $t_0 \not\equiv t_1$ we must require that the labels match, $l_0 = l_1$, and that $\xrightarrow{l_0}$ is deterministic. We require the labels to match because if they do not, then given only $p \xrightarrow{l} p'$ it is impossible to prove that $f(p,p)$ can simulate it using only a $2^*_{l,l'}$ rule. The determinacy of the transition with that label is necessary when proving that transitions from $f(p,p)$ can, in general, be simulated by p; if we assume that $f(p,p) \xrightarrow{l} p'$ then we must be able to conclude that p' has the shape $f(p'',p'')$ for some p'', in order to meet the bisimulation condition for $\simeq_f$. Consider the standard choice operator $+$ and prefixing operator $.$ of CCS with the $|$ operator from the last example, with $\alpha \in \{a,b,c\}$.

$$\frac{}{\alpha \xrightarrow{\alpha} 0} \quad \frac{}{\alpha.x \xrightarrow{\alpha} x} \quad \frac{x \xrightarrow{\alpha} x'}{x+y \xrightarrow{\alpha} x'} \quad \frac{y \xrightarrow{\alpha} y'}{x+y \xrightarrow{\alpha} y'} \quad \frac{x \xrightarrow{\alpha} x' \quad y \xrightarrow{\alpha} y'}{x|y \xrightarrow{\alpha} x'|y'}$$

If we let $p = a.b + a.c$, then $p|p \xrightarrow{a} b|c$ and $b|c$ is stuck. However, p cannot simulate this transition w.r.t. $\simeq_f$. Indeed, p and $p|p$ are not bisimilar.

4.3 Predicates

There are many examples of TSSs where predicates are used. The definitions presented in Section 2 and 4 can be easily adapted to deal with predicates as well. In particular, two closed terms are called bisimilar in this setting when, in addition to the transfer conditions of bisimilarity, they satisfy the same predicates. To extend the idempotency rule format to a setting with predicates, the following types of rules for predicates are introduced:

3_P. *Choice rules for predicates*

$$\frac{\{Px_i\} \cup \Phi}{Pf(x_0,x_1)}, \quad i \in \{0,1\}$$

4_P. *Synchronization rules for predicates*

$$\frac{\{Px_0,\, Px_1\} \cup \Phi}{Pf(x_0,x_1)}$$

As before, we define the starred version of these forms, 3^*_P and 4^*_P. The starred version of each rule is the same as the unstarred one except that the set Φ must be empty in each case. With these additional definition the idempotency format is defined as follows.

A TSS is in *idempotency format w.r.t. a binary operator* f if each f-defining rule, i.e., a deduction rule with f appearing in the source of the conclusion, is of

one the forms 1_l, $2_{l_0,l_1}$, 3_P or 4_P for some $l, l_0, l_1 \in L$, for each label $l \in L$ and predicate symbol P. Moreover, for each $l \in L$, there exists at least one rule of the forms 1^*_l or $2^*_{l,l}$, and for each predicate symbol P there is a rule of the form 1^*_P or 2^*_P.

4.4 Examples

Example 6. The most prominent example of an idempotent operator is non-deterministic choice, denoted $+$. It typically has the following deduction rules:

$$\frac{x_0 \xrightarrow{a} x_0'}{x_0 + x_1 \xrightarrow{a} x_0'} \qquad \frac{x_1 \xrightarrow{a} x_1'}{x_0 + x_1 \xrightarrow{a} x_1'}$$

Clearly, these are in the idempotency format w.r.t. $+$.

Example 7 (External choice). The well-known external choice operator $\Box$ from CSP [11] has the following deduction rules

$$\frac{x_0 \xrightarrow{a} x_0'}{x_0 \Box x_1 \xrightarrow{a} x_0'} \qquad \frac{x_1 \xrightarrow{a} x_1'}{x_0 \Box x_1 \xrightarrow{a} x_1'} \qquad \frac{x_0 \xrightarrow{\tau} x_0'}{x_0 \Box x_1 \xrightarrow{\tau} x_0' \Box x_1} \qquad \frac{x_1 \xrightarrow{\tau} x_1'}{x_0 \Box x_1 \xrightarrow{\tau} x_0 \Box x_1'}$$

Note that the third and fourth deduction rule are not instances of any of the allowed types of deduction rules. Therefore, no conclusion about the validity of idempotency can be drawn from our format. In this case this does not point to a limitation of our format, because this operator is not idempotent in strong bisimulation semantics [7].

Example 8 (Strong time-deterministic choice). The choice operator that is used in the timed process algebra ATP [17] has the following deduction rules.

$$\frac{x_0 \xrightarrow{a} x_0'}{x_0 \oplus x_1 \xrightarrow{a} x_0'} \qquad \frac{x_1 \xrightarrow{a} x_1'}{x_0 \oplus x_1 \xrightarrow{a} x_1'} \qquad \frac{x_0 \xrightarrow{\chi} x_0' \quad x_1 \xrightarrow{\chi} x_1'}{x_0 \oplus x_1 \xrightarrow{\chi} x_0' \oplus x_1'}$$

The idempotency of this operator follows from our format since the last rule for $\oplus$ fits the form $2^*_{\chi,\chi}$ because, as we remarked in Example 4, the transition relation $\xrightarrow{\chi}$ is deterministic.

Example 9 (Weak time-deterministic choice). The choice operator $+$ that is used in most ACP-style timed process algebras has the following deduction rules:

$$\frac{x_0 \xrightarrow{a} x_0'}{x_0 + x_1 \xrightarrow{a} x_0'} \qquad \frac{x_1 \xrightarrow{a} x_1'}{x_0 + x_1 \xrightarrow{a} x_1'}$$

$$\frac{x_0 \xrightarrow{1} x_0' \quad x_1 \xrightarrow{1} x_1'}{x_0 + x_1 \xrightarrow{1} x_0' + x_1'} \qquad \frac{x_0 \xrightarrow{1} x_0' \quad x_1 \xrightarrow{1} \not\to}{x_0 + x_1 \xrightarrow{1} x_0'} \qquad \frac{x_0 \xrightarrow{1} \not\to \quad x_1 \xrightarrow{1} x_1'}{x_0 + x_1 \xrightarrow{1} x_1'}$$

The third deduction rule is of the form $2^*_{1,1}$, the others are of forms 1^*_a and 1^*_1. This operator is idempotent (since the transition relation $\xrightarrow{1}$ is deterministic, as remarked in Example 5).

Example 10 (Conjunctive nondeterminism). The operator $\vee$ as defined in Example 2 by means of the deduction rules

$$\frac{x \ \mathrm{can}_a}{x \vee y \ \mathrm{can}_a} \qquad \frac{y \ \mathrm{can}_a}{x \vee y \ \mathrm{can}_a} \qquad \frac{x \ \mathrm{after}_a \ x' \quad y \ \mathrm{after}_a \ y'}{x \vee y \ \mathrm{after}_a \ x' \vee y'}$$

satisfies the idempotency format (extended to a setting with predicates). The first two deduction rules are of the form $3^*_{\mathrm{can}_a}$ and the last one is of the form $2^*_{a,a}$. Here we have used the fact that the transition relations after_a are deterministic as concluded in Example 2.

Example 11 (Delayed choice). Delayed choice can be concluded to be idempotent in the restricted setting without $+$ and $\cdot$ by using the idempotency format and the fact that in this restricted setting the transition relations $\xrightarrow{a}$ are deterministic. (See Example 3.)

$$\frac{x\downarrow}{x \mathbin{\overline{+}} y\downarrow} \qquad \frac{y\downarrow}{x \mathbin{\overline{+}} y\downarrow} \qquad \frac{x \xrightarrow{a} x' \quad y \xrightarrow{a} y'}{x \mathbin{\overline{+}} y \xrightarrow{a} x' \mathbin{\overline{+}} y'} \qquad \frac{x \xrightarrow{a} x' \quad y \not\xrightarrow{a}}{x \mathbin{\overline{+}} y \xrightarrow{a} x'} \qquad \frac{x \not\xrightarrow{a} \quad y \xrightarrow{a} y'}{x \mathbin{\overline{+}} y \xrightarrow{a} y'}$$

The first two deduction rules are of form $3^*_{\downarrow}$, the third one is a $2^*_{a,a}$ rule, and the others are 1_a rules. Note that for any label a starred rule is present.

For the extensions discussed in Example 3 idempotency cannot be established using our rule format since the transition relations are no longer deterministic. In fact, delayed choice is not idempotent in these cases.

5 Conclusions

In this paper, we presented two rule formats guaranteeing determinism of certain transitions and idempotency of binary operators. Our rule formats cover all practical cases of determinism and idempotency that we have thus far encountered in the literature.

We plan to extend our rule formats with the addition of data/store. Also, it is interesting to study the addition of structural congruences pertaining to idempotency to the TSSs in our idempotency format.

References

1. L. Aceto, W.J. Fokkink, and C. Verhoef. Structural operational semantics. In *Handbook of Process Algebra, Chapter 3*, pages 197–292. Elsevier, 2001.
2. J.C.M. Baeten and C.A. Middelburg. *Process Algebra with Timing*. EATCS Monographs. Springer-Verlag, Berlin, Germany, 2002.
3. J.C.M. Baeten and W.P. Weijland. *Process Algebra*, volume 18 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1990.
4. J.C.M. Baeten and S. Mauw. Delayed choice: An operator for joining Message Sequence Charts. In *Proceedings of Formal Description Techniques*, volume 6 of *IFIP Conference Proceedings*, pages 340–354. Chapman & Hall, 1995.

5. J.A. Bergstra and J.W. Klop. Process algebra for synchronous communication. *Information and Control*, 60(1-3):109–137, 1984.
6. S. Cranen, M.R. Mousavi, and M.A. Reniers. A rule format for associativity. In *Proceedings of CONCUR'08*, volume 5201 of *Lecture Notes in Computer Science*, pages 447–461, Springer-Verlag, 2008.
7. P.R. D'Argenio. τ-angelic choice for process algebras (revised version). Technical report, LIFIA, Depto. de Informática, Fac. de Cs. Exactas, Universidad Nacional de La Plata. March 1995.
8. W.J. Fokkink and T. Duong Vu. Structural operational semantics and bounded nondeterminism. *Acta Informatica*, 39(6–7):501–516, 2003.
9. J.F. Groote. Transition system specifications with negative premises. *Theoretical Computer Science (TCS)*, 118(2):263–299, 1993.
10. M. Hennessy and G.D. Plotkin. Finite conjuncitve nondeterminism. In *Concurrency and Nets, Advances in Petri Nets*, pages 233–244. Springer, 1987.
11. C.A.R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.
12. R. Lanotte and S. Tini. Probabilistic congruence for semistochastic generative processes. In *Proceedings of FOSSACS'05*, volume 3441 of *Lecture Notes in Computer Science*, pages 63–78. Springer, 2005.
13. M.R. Mousavi, I.C.C. Phillips, M.A. Reniers, and I. Ulidowski. The meaning of ordered SOS. In *Proceedings of FSTTCS'06*, volume 4337 of *Lecture Notes in Computer Science*, pages 334–345, Springer, 2006.
14. M.R. Mousavi, M.A. Reniers, and J.F. Groote. A syntactic commutativity format for SOS. *Information Processing Letters*, 93:217–223, 2005.
15. M.R. Mousavi and M.A. Reniers. Orthogonal extensions in structural operational semantics. In *Proceedings of the ICALP'05*, volume 3580 of *Lecture Notes in Computer Science*, pages 1214–1225. Springer, 2005.
16. M.R. Mousavi, M.A. Reniers, and J.F. Groote. SOS formats and meta-theory: 20 years after. *Theoretical Computer Science*, (373):238–272, 2007.
17. X. Nicollin and J. Sifakis. The algebra of timed processes ATP: Theory and application. *Information and Computation*, 114(1):131–178, 1994.
18. G.D. Plotkin. A structural approach to operational semantics. *Journal of Logic and Algebraic Progamming*, 60:17–139, 2004. This article first appeared as Technical Report DAIMI FN-19, Computer Science Department, Aarhus University.
19. S. Tini. Rule formats for compositional non-interference properties. *Journal of Logic and Algebraic Progamming*, 60:353–400, 2004.
20. I. Ulidowski and S. Yuen. Extending process languages with time. In *Proceedings of AMAST'97*, volume 1349 of *Lecture Notes in Computer Science*, pages 524–538. Springer, 1997.
21. M. van Weerdenburg and M.A. Reniers. Structural operational semantics with first-order logic. In *Pre-proceedings of SOS'08*, pages 48–62, 2008.
22. C. Verhoef. A congruence theorem for structured operational semantics with predicates and negative premises. *Nordic Journal of Computing*, 2(2):274–302, 1995.