Orthogonal Extensions in Structural Operational Semantics

(Extended Abstract)

MohammadReza Mousavi and Michel A. Reniers

Department of Computer Science,
Eindhoven University of Technology,
NL-5600MB Eindhoven, The Netherlands

Abstract. In this paper, we give novel and more liberal notions of operational and equational conservativity for language extensions. We motivate these notions by showing their practical application in existing formalisms. Based on our notions, we formulate and prove meta-theorems that establish conservative extensions for languages defined using Structural Operational Semantics (SOS).

Key words: Formal Semantics, Structural Operational Semantics (SOS), Conservative Extension, Operational Conservativity, Equational Conservativity, Orthogonality.

1 Introduction

Programming languages and process calculi have been subject to constant extensions. It is often crucial to make sure that such extensions do not change the intuition behind the old subset, or said otherwise, the extensions are *conservative*. In the context of languages with Structural Operational Semantics (SOS) [13], this topic has been studied in depth in [1, 3, 5, 10, 15, 17]. This research has resulted in meta-theorems proving sufficient conditions for an extension to be *operationally* and/or *equationally conservative*. In the remainder, we mostly refer to [5] which gives the most detailed account of the problem and subsumes almost all previous results. We do not treat multi-sorted and variable binding signatures, addressed in [5, 10], in this paper.

So far, *operational conservativity* has only allowed for extensions that consistently deny the addition of any new behavior to the old syntax. One can imagine that an extension which grants a new behavior consistently to the old syntax can also be considered safe or "conservative". This phenomenon occurs quite often in practice. For example, designers of many timed extensions of existing formalisms (e.g., the timed process algebras of [2, 9, 14]) have decided to add timed behavior homogenously to the terms from the old syntax. Unfortunately, it turns out that the existing definitions and their corresponding meta-theorems come short of any formal result about such extensions.

In this paper, we present a more liberal notion of operational conservativity, called *orthogonality*, which caters for both possibilities (i.e., denying some types of behavior from the old syntax while granting some other types). We show that our notion is useful in the aforementioned cases where the old notions cannot be used. We formulate orthogonality meta-theorems for languages with Structural Operational Semantics and prove them correct.

In [15], *equational conservativity* is considered in the setting where a new set of axioms is added to an existing set. Then, the extension is called *equationally conservative* if it induces exactly the same derivable closed equalities on the old syntax as the original equational theory. In this paper, we remove the requirement for including the old set of axioms in the extended equational theory. This relaxation is motivated by the fact that in many extensions, such as those of [2, 14], for some axioms, only all closed derivable equalities on the old syntax are kept and the axioms themselves are removed. Operational conservativity is usually considered as a means for equational conservativity and we show that our notion of orthogonality leads to equational conservativity in the same way as operational conservativity does (no matter which notion of equational conservativity is chosen, the traditional notion or the relaxed one).

The rest of this paper is structured as follows. Section 2 gives the basic definitions about Structural Operational Semantics, Transition System Specification (TSS) and equational theory. Section 3 presents the notions of operational and equational conservativity and gives sufficient conditions for proving operational conservativity. Orthogonality and related notions are defined in Section 4. Subsequently, Section 5 defines sufficient conditions for orthogonality. In the same section, we also present theorems establishing the link between orthogonality and equational conservativity. Finally, Section 6 summarizes the results and presents future directions. In each section, we provide abstract and concrete examples from the area of process algebra to motivate the definitions and illustrate the results. Due to lack of space, we could not present all of the results and the proofs of the theorems in this extended abstract. Interested readers can find these in the full version of this paper [11].

2 Preliminaries

2.1 Structural Operational Semantics

Structural Operational Semantics [13] is a logical way of defining operational semantics which has found lots of applications in different areas of computer science. A semantic specification in the style of SOS, called a *Transition System Specification (TSS)*, consists of a number of deduction rules which specify the possibility of a transition (in the conclusion of the rules) in terms of the (im)possibility of other transition (in the premises). Predicates on states are other possible ingredients of TSS's which can both be defined in the conclusion of the rules and used in the premises. Predicates can always be coded as transitions with dummy right-hand sides (cf. [16]) and thus, we do not complicate

the presentation with their formal treatment. Next, we formalize the rest of the concepts mentioned above.

Definition 1 (Term and Substitution) We assume that the set of *process terms*, denoted by $T(\Sigma)$ with typical members $t, t', t_i, \ldots$, is inductively defined on a given set of *variables* $V = \{x, y, \ldots\}$ and a signature Σ. A *signature* contains a number of *function symbols* (composition operators: $f, g, \ldots$) with fixed *arities*. Function symbols with arity 0 are called *constants*. *Closed terms*, denoted by $C(\Sigma)$ with typical members $p, q, p_i, \ldots$, are terms that do not contain variables. The set of variables appearing in term t is denoted by $vars(t)$.

A *(closed) substitution* σ replaces variables in a term with other (closed) terms. *The set of terms generated by a set of terms* S, denoted by $G(S)$, is the set of all terms $t' = \sigma(t)$, for some $t \in S$ and some σ such that $\forall_{x \in V} \sigma(x) \in S$. A set of terms S *covers* Σ-terms, if $C(\Sigma) \subseteq G(S)$.

A transition system specification, defined below, is a logical way of defining a transition relation on (closed) terms.

Definition 2 (Transition System Specification (TSS)) A *transition system specification* is a tuple (Σ, L, D) where Σ is a signature, L is a set of *labels* (with typical members $l, l', l_0, \ldots$) and D is a set of deduction rules. For all $l \in L$, and $t, t' \in T(\Sigma)$ we define that $(t, l, t') \in \rightarrow$ and $(t, l) \notin \rightarrow$ are *formulae* (positive and negative, respectively). To avoid any confusion, note that $\ldots \in \rightarrow$ and $\ldots \notin \rightarrow$ are used as a syntactic notation and are not intended to denote the set-theoretic membership at this point. The notion of closed is lifted from terms to formulae in the natural way. A *deduction rule* $dr \in D$, is defined as a tuple (H, c) where H is a set of formulae and c is a positive formula. The formula c is called the *conclusion* and the formulae from H are called *premises*. A deduction rule with label l in its conclusion is called an *l-rule*.

Formulae $(t, l, t') \in \rightarrow$ and $(t, l) \notin \rightarrow$ are denoted by the more intuitive notations $t \xrightarrow{l} t'$ and $t \stackrel{l}{\not\rightarrow}$, respectively. We refer to t as the source of both formulae and to t' as the target of the first one. A deduction rule (H, c) is denoted by $\frac{H}{c}$ in the remainder.

Different interpretations of the transition relation (the set of closed positive formulae) induced by a TSS are given in the literature. In [6], an extensive overview of alternative interpretations is provided. We formulate and prove our main results in such a general way that they remain independent from the chosen interpretation and can be adopted for several existing ones. In cases where we need an explicit transition relation, we assume that this transition relation is uniquely defined by the corresponding TSS using one of the interpretations given in [6]. In such cases, we use the notation $tss \models \phi$ to denote that a closed positive formula ϕ is in the transition relation induced by tss.

One criterium that guarantees the existence and uniqueness of a transition relation associated with a TSS is the following concept of (strict) stratification, which we use for other purposes in this paper, as well.

Definition 3 (Stratification [8]) A TSS tss is stratified by a function $\mathcal{S}$ from closed positive formulae to an ordinal if and only if for all deduction rules in tss of the following form:

$$\frac{\{t_i \xrightarrow{l_i} t'_i | i \in I\} \quad \{t_j \xrightarrow{l_j} | j \in J\}}{t \xrightarrow{l} t'}$$

and for all closed substitutions σ, $\forall_{i \in I} \mathcal{S}(\sigma(t_i \xrightarrow{l_i} t'_i)) \leq \mathcal{S}(\sigma(t \xrightarrow{l} t'))$ and $\forall_{j \in J}$ $\mathcal{S}(\sigma(t_j \xrightarrow{l_j} t'')) < \mathcal{S}(\sigma(t \xrightarrow{l} t))$, for all terms t''. If the measure decreases also from the conclusion to the positive premises, then tss is *strictly stratified* by $\mathcal{S}$.

The following example illustrates the concepts defined in this section.

Example 1 (Minimal Process Algebra (*MPA*)) Consider the following deduction rules defined on a signature with a constant δ, a family of unary operators $a._-$ (for all $a \in A$, where A is a given set of atomic actions) and a binary operator $\_ + \_$. The labels of transitions are $a \in A$.

$$\textbf{(a)} \frac{}{a.x \xrightarrow{a} x}(a \in A) \quad \textbf{(c0)} \frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \quad \textbf{(c1)} \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$$

This TSS (called tss_m in the remainder) is supposed to define a transition relation for the Minimal Process Algebra (*MPA*) of [2], simplified here by removing the concept of termination, which we use as our running example in the remainder. Deduction rules of *MPA* are (strictly) stratified using a measure of size on the terms in the source of formulae and it defines a unique transition relation by all possible interpretations. The following transitions are among those: $tss_m \vDash (a.\delta) + \delta \xrightarrow{a} \delta$ and $tss_m \vDash a.(\delta + a.\delta) \xrightarrow{a} \delta + a.\delta$.

2.2 Equational Theory

Equational theories play a central role in process algebras. They capture the basic intuition behind the algebra, and the models of the algebra are expected to respect this intuition (e.g., the models induced by operational semantics). To establish a reasonable link between the operational model and the equational theory of the algebra, a notion of behavioral equality is needed. This notion captures when two syntactic terms show the "same behavior" and thus they should belong to the same equivalence class. There is a spectrum of notions of behavioral equality in the literature [7]. We take the notion of strong bisimilarity [12], denoted by $\leftrightarrow$, as the notion of behavioral equivalence, but as we show in the extended version of this paper [11], our results are valid for a wide range of notions in this spectrum.

Getting back to the equational side of the algebra, the notion of behavioral equivalence should ideally coincide with the closed derivations of the equational theory. One side of this coincidence is captured by the soundness theorem which

states that all closed derivations of the equational theory are indeed valid with respect to the particular notion of behavioral equality. The other side of the coincidence, called completeness, phrases that all induced behavioral equalities are derivable from the equational theory, as well. These concepts are formalized in the remainder.

Definition 4 (Equational Theory) An *equational theory* or *axiomatization* (Σ, E) is a set of equalities E on a signature Σ of the form $t = t'$, where $t, t' \in T(\Sigma)$. A closed instance $p = p'$, for some $p, p' \in C(\Sigma)$, is derivable from E, denoted by $E \vdash p = p'$ if and only if it is in the smallest congruence relation induced by the equalities of E.

An equational theory (Σ, E) is *sound* with respect to a TSS tss (also on signature Σ) if and only if for all $p, p' \in C(\Sigma)$, if $E \vdash p = p'$, then it holds that $tss \vdash p \leftrightarrows p'$. It is *complete* if the implication holds in the other direction.

An equational theory E on Σ *eliminates* function symbols from $\Sigma' \subseteq \Sigma$ if and only if for all $p \in C(\Sigma)$ there exists a term $p' \in C(\Sigma \setminus \Sigma')$ such that $E \vdash p = p'$.

The following example illustrates the idea of equational theory.

Example 2 (*MPA*: Equational Theory) Consider the Minimal Process Algebra of Example 1. The following is an axiomatization of *MPA* [2].

$$x + y = y + x \qquad x + (y + z) = (x + y) + z \qquad x + x = x \qquad x + \delta = x$$

It is well-known that this axiomatization is sound and complete with respect to tss_m given in Example 1. The following are examples of derivable equalities from the above axiomatization: $(a.\delta) + \delta = a.\delta$ and $(a.\delta) + a.\delta = a.\delta$.

3 Operational and Equational Conservativity

In this section, we define different concepts regarding language extensions. To extend a language defined by a TSS, one may have to combine an existing signature with a new one. However, not all signatures can be combined into one as the arities of the function symbols may clash. To prevent this, we define two signatures to be *consistent* when they agree on the arity of the shared function symbols. Henceforth, we always assume that extended and extending TSS's are consistent. The following definition formalizes the concept of operational extension.

Definition 5 (Extension of a TSS) Consider TSS's $tss_0 = (\Sigma_0, L_0, D_0)$ and $tss_1 = (\Sigma_1, L_1, D_1)$. The extension of tss_0 with tss_1, denoted by $tss_0 \cup tss_1$, is defined as $(\Sigma_0 \cup \Sigma_1, L_0 \cup L_1,\ D_0 \cup D_1)$.

Next, we define when an extension of a TSS is called operationally conservative.

Definition 6 (Operational Conservativity [15]) Consider TSS's $tss_0 = (\Sigma_0, L_0, D_0)$ and $tss_1 = (\Sigma_1, L_1, D_1)$. If $\forall_{p \in C(\Sigma_0)}\ \forall_{p' \in C(\Sigma_0 \cup \Sigma_1)}\ \forall_{l \in L_0 \cup L_1}\ tss_0 \cup tss_1 \vDash p \xrightarrow{l} p'$ $\Leftrightarrow tss_0 \vDash p \xrightarrow{l} p'$, then $tss_0 \cup tss_1$ is an *operationally conservative extension* of tss_0.

Note that in the above definition, the labels and the targets of the transitions are taken from the extended TSS and thus, any new transition of the old syntax, even with a new label or a new target is prohibited. The following example illustrates the idea of extending TSS's and the concept of operational conservativity.

Example 3 (Timed *MPA*: Operational Semantics) Consider the following deduction rules (divided into three parts) which are defined on a signature with two constants δ and $\underline{\delta}$, a unary function symbol $\underline{\underline{\sigma}}._-$, two families of unary function symbols $a._-$ and $\underline{\underline{a}}._-$ (for all $a \in A$) and a binary function symbol $\_ + \_$. The set of labels of the TSS is $A \cup \{1\}$ (for $1 \notin A$).

(1) $$\textbf{(ua)}\frac{}{\underline{\underline{a}}.x \xrightarrow{a} x} \qquad \textbf{(td)}\frac{}{\underline{\underline{\sigma}}.x \xrightarrow{1} x}$$

(2) $$\textbf{(tc0)}\frac{x \xrightarrow{1} x' \quad y \xrightarrow{1} y'}{x + y \xrightarrow{1} x' + y'} \qquad \textbf{(tc1)}\frac{x \xrightarrow{1} x' \quad y \overset{1}{\not\rightarrow}}{x + y \xrightarrow{1} x'} \qquad \textbf{(tc2)}\frac{y \xrightarrow{1} y' \quad x \overset{1}{\not\rightarrow}}{x + y \xrightarrow{1} y'}$$

(3) $$\textbf{(ta)}\frac{}{a.x \xrightarrow{1} a.x} \qquad \textbf{(d)}\frac{}{\delta \xrightarrow{1} \delta}$$

The above TSS, which we call tss_t defines the aspect of timing in terms of new time transitions $\xrightarrow{1}$ and it is added in [2] to tss_m in Example 1 to define a relative-discrete-time extension of *MPA*. The intuition behind the new underlined function symbols ($\underline{\underline{a}}._-$ and $\underline{\underline{\sigma}}._-$) is that they are not delayable in time and should take their (respectively action and time) transitions immediately. Addition of the first and/or the second parts of the above TSS (each or both) to tss_m results in an operationally conservative extension of the latter as the newly added transitions will be restricted to the new syntax. (Note that in the first and second parts, there is no rule about timed transition of constants in the old syntax.) This claim can be checked formally as an instance of a meta-theorem in the rest of this section. However, the addition of part (3) violates the conservativity of the extension as it adds time transitions ($\xrightarrow{1}$) to the behavior of terms from the old syntax. For example, in combination with the first two parts, it allows for transitions such as $tss_m \cup tss_t \vDash a.\delta \xrightarrow{1} a.\delta$ and $tss_m \cup tss_t \vDash (a.\delta) + \delta \xrightarrow{1} (a.\delta) + \delta$, all of which are prohibited by the original TSS and thus are considered harmful from the operational conservativity point of view.

Next, we formulate sufficient conditions to prove operational conservativity. But before that, we need a few auxiliary definitions.

Definition 7 (Source Dependency) All variables appearing in the source of the conclusion of a deduction rule are called *source dependent*. A variable of a deduction rule is *source dependent* if it appears in a target of a premise of which all

the variables of the source are source dependent. A formula is *source dependent* when all the variables appearing in it are source dependent. A deduction rule is *source dependent* when all its variables are. A TSS is *source dependent* when all its rules are.

Definition 8 (Reduced Rules) For a deduction rule $d = (H, c)$, the reduced rule with respect to a signature Σ is defined by $\rho(d, \Sigma) \doteq (H', c)$ where H' is the set of all premises from H which have a Σ-term as a source.

Theorem 1 (Operational Conservativity Meta-Theorem [5]) Given two TSS's $tss_0 = (\Sigma_0, L_0, D_0)$ and $tss_1 = (\Sigma_1, L_1, D_1)$, $tss_0 \cup tss_1$ is an operationally conservative extension of tss_0 if:

1. tss_0 is source dependent;
2. for all $d \in D_1$ at least one of the following holds:
   (a) the source of the conclusion has a function symbol in $\Sigma_0 \setminus \Sigma_1$, or
   (b) $\rho(d, \Sigma_0)$ has a source-dependent positive premise $t \xrightarrow{l} t'$ such that $l \notin \Sigma_0$ or $t' \notin T(\Sigma_0)$.

The following definition formalizes the concept of equational conservativity.

Definition 9 (Equational Conservativity) An equational theory E_1 on signature Σ_1 is an equationally conservative extension of E_0 on Σ_0 if and only if $\Sigma_0 \subseteq \Sigma_1$ and for all $p, p' \in C(\Sigma_0)$, $E_0 \vdash p = p' \Leftrightarrow E_1 \vdash p = p'$.

It is worth mentioning that the above definition is more liberal than the similar notion in [15] in that there, it is required that the same axioms are included in the extended equational theory (i.e., $E_0 \subseteq E_1$). In practice, some process algebras do not keep the same axioms when extending the formalism while they make sure that the closed instantiations of the old axioms with old terms indeed remain derivable (see for example, [2, 14] and Example 4 in the remainder). Hence, we believe that the restriction imposed by [15] unnecessarily limits the applicability of the theory. If, for any reason, one chooses the more restricted notion of [15], the theorems concerning equational conservativity in this paper remain valid.

Example 4 (Timed *MPA*: Equational Theory) Consider the TSS resulting from extending tss_m of Example 1 with (all three parts) of the timed extension defined in Example 3. The following are a set of sound and complete axioms (w.r.t. strong bisimilarity) for this TSS:

$$x + y = y + x \quad x + (y + z) = (x + y) + z \qquad x + x = x \qquad \delta = \underline{\sigma}.\delta$$

$$x + \underline{\delta} = x \qquad (\underline{\sigma}.x) + \underline{\sigma}.y = \underline{\sigma}.(x + y) \quad a.x = (\underline{a}.x) + \underline{\sigma}.a.x \quad (a.x) + \delta = a.x$$

The above axiomatization underscores the fact we mentioned before. Namely, the axioms of the old system do not hold in the new system (e.g., $(\underline{a}.x) + \delta \neq \underline{a}.x$ as an instance of $x + \delta = x$) but all closed instantiations of the old axioms by the old syntax are derivable from the new set of axioms.

It can be checked that the above axiomatization of timed *MPA* is indeed an equationally conservative extension of the axiomatization of *MPA* in the sense of Definition 9. Thus, if one considers operational conservativity as a means to equational conservativity, this example already suggests the need for an extension of Definition 6. In other words, we believe that the transitions added by the extension are quite innocent and harmless to the intuition behind the original semantics, for they are added uniformly to the old syntax without changing the old behavior or violating previously valid equalities. In the next section, we formalize our idea of orthogonal extensions which caters for extensions of the above type.

4 Orthogonality

In this section, we define the notion of orthogonality and an instance of this notion, called *granting extensions*, which can be checked syntactically.

Definition 10 (Orthogonal Extension) Consider TSS's $tss_0 = (\Sigma_0, L_0, D_0)$ and $tss_1 = (\Sigma_1, L_1, D_1)$. The TSS $tss_0 \cup tss_1$ is an *orthogonal* extension of tss_0 when first, $\forall_{p,p' \in C(\Sigma_0)}\ \forall_{l \in L_0}\ tss_0 \cup tss_1 \vDash p \xrightarrow{l} p' \Leftrightarrow tss_0 \vDash p \xrightarrow{l} p'$ and second, and $\forall_{p,p' \in C(\Sigma_0)}\ tss_0 \cup tss_1 \vDash p \leftrightarrows p' \Leftrightarrow tss_0 \vDash p \leftrightarrows p'$.

Note that it immediately follows from the above definition that orthogonality is a preorder, i.e., a reflexive and transitive relation, on TSS's. Besides strong bisimilarity, our results in the this paper are valid for orthogonality with respect to most other notions of behavioral equivalence in the literature (cf. [11]). The notion of *operational conservativity up to ϕ-equivalence* of [15, 3] can be seen as a variant of orthogonality which only has the second condition. This and other variants of the notion of orthogonality can also be useful and our results can be used to establish meta-theorems for these notions. To our knowledge, beyond operational conservativity results (e.g., in [15]), no systematic study of these notions (including meta-theorems guaranteeing them) has been carried out.

Corollary 1 An operationally conservative extension is an orthogonal extension.

Corollary 1 addresses operational conservativity as an extreme case of orthogonality which denies all new transitions from the old syntax; the other extreme is an extension which grants all new behavior to the old syntax. However, for such an extension to be orthogonal, these transitions should be made to equivalent terms from the old syntax. In particular, if we allow for self transitions, we are able to prove orthogonality with respect to many notions of behavioral equivalence. The following definitions and the subsequent theorem substantiate these concepts.

Definition 11 (Granting Extension) Consider TSS's $tss_0 = (\Sigma_0, L_0, D_0)$ and $tss_1 = (\Sigma_1, L_1, D_1)$ with disjoint labels. We call $tss_0 \cup tss_1$ a *granting extension*

of tss_0 when first, $\forall_{p,p' \in C(\Sigma_0)}\ \forall_{l \in L_0}\ tss_0 \vDash p \xrightarrow{l} p' \Leftrightarrow tss_0 \cup tss_1 \vDash p \xrightarrow{l} p'$ and second, $\forall_{p \in C(\Sigma_0)}\ \forall_{p' \in C(\Sigma_0 \cup \Sigma_1)}\ \forall_{l \in L_1 \setminus L_0}\ tss_0 \cup tss_1 \vDash p \xrightarrow{l} p' \Leftrightarrow p = p'$.

The above definition states that granting extensions keep the old transitions on the old terms intact and only add self transitions with all of the new labels to old terms. This definition does not make any statement about the transitions on the new terms, i.e., terms from $T(\Sigma_0 \cup \Sigma_1) \setminus T(\Sigma_0)$.

We are doubtful whether any meaningful relaxation of Definition 11 would be at all possible that allows for anything coarser than syntactic equality on the old terms involved in (the left- or the right-hand side of) the new transitions and still can be captured by simple syntactic checks. This suggests that to formulate syntactic criteria for proving orthogonality, we have to resort to one of the two extremes (operational conservativity or granting extensions). We admit that combining these two extremes is interesting. This is partly possible by exploiting the the transitivity of orthogonal extension relation. This way, one can interleave the application of granting and operational conservativity theorems (cf. Example 5). We propose an alternative method of combining operationally conservative and granting extensions in [11]. Next, we show that granting extensions are indeed orthogonal.

Theorem 2 For TSS's tss_0 and tss_1, if tss_1 is a granting extension of tss_0 then tss_1 is an orthogonal extension of tss_0.

5 Meta-Theorems

In this section, we seek sufficient conditions for establishing orthogonality and equational conservativity.

We start with defining sufficient conditions to prove an extension to be granting. Hence, we need to define when a deduction rule proves (only) self transitions. We use unification as a means to this end.

Definition 12 (Unification) A term t is *unifiable* with t' using σ, denoted by $t \approx_\sigma t'$ if and only if $\sigma(t) = \sigma(t')$. The set of unifiers of t and t' is defined by $U(t, t') = \{\sigma \mid t \approx_\sigma t'\}$. The set of unifiers of a set of pairs is defined as the intersection of the sets of unifiers of each pair. The set of unifiers of an empty set is defined to include all substitutions.

The set of unifiers of a positive formula $t \xrightarrow{l} t'$ is defined as the set of unifiers of t and t'. Unification also naturally extends to a set of positive formulae, again, using intersection.

Next, we characterize the set of rules that induce self transitions. This is done by only allowing for unifiable (positive) formulae in the premises and the conclusion of a rule and further, by forcing the unification of the conclusion to follow from that of the premises.

Definition 13 (Source Preserving Rules) A deduction rule $\frac{H}{c}$ without negative premises is *source preserving* if $U(H) \neq \emptyset$ and $U(H) \subseteq U(c)$. A TSS is *source preserving* if all its deduction rules are. For a source preserving TSS, the set of *unified conclusions* contains conclusions of the deduction rules with their unifiers applied to them.

Source-preserving rules are safe for the purpose of proving self transitions. However, there might be other rules in the extending TSS that can be harmful in that they may prove other types of transition for old terms. This may be prevented by forcing the other (non-source-preserving) rules to have negative or non-unifiable positive premises addressing the old syntax. The following definition gives sufficient conditions for an extension to be granting.

Definition 14 (Granting Criteria) Consider a TSS $tss = (\Sigma, L, D)$ stratified by $\mathcal{S}$. It *grants* L_0 *transitions on* Σ_0*-terms*, if $tss = tss_0 \cup tss_1$ (with $tss_x = (\Sigma_x, L_x, D_x)$ for $x \in \{0, 1\}$) such that:

1. tss_0 is strictly stratified by $\mathcal{S}$, it is source preserving and for all $l \in L_0$, the set containing sources of unified conclusions of l-rules covers Σ_0-terms, and
2. for all deduction rules $d \in D_1$ at least one of the following holds:
   (a) d has a function symbol from $\Sigma_1 \setminus \Sigma_0$ in the source of its conclusion, or
   (b) $\rho(d, \Sigma_0)$ has a negative source-dependent premise with a label in L_1, or
   (c) $\rho(d, \Sigma_0)$ has a positive source-dependent premise $t \xrightarrow{l} t'$ with $l \in L_1$ and $U(t, t') = \emptyset$.

The first condition in the above definition is dedicated to proving self transitions from the syntax of Σ_0, and the the second one takes care of preventing Σ_0-terms from performing other types of transitions while allowing other terms to do so.

Theorem 3 (Granting Meta-theorem) Consider source-dependent TSS's $tss_0 = (\Sigma_0, L_0, D_0)$ and $tss_1 = (\Sigma_1, L_1, D_1)$. If tss_1 grants L_1 transitions on Σ_0-terms and $L_0 \cap L_1 = \emptyset$ then $tss_0 \cup tss_1$ is a granting extension of tss_0.

The following example applies our meta-theorem to obtain orthogonality of relative-discrete-time extension of *MPA*.

Example 5 (Timed *MPA*: Orthogonality) Consider the tss_m of *MPA* in Example 1 and tss_t of Example 3. TSS tss_t can be decomposed into the following three parts: $tss_0 \doteq (\{\underline{a}._-, \underline{\delta}\}, A, \{\textbf{(ua)}, \textbf{(td)}\})$, $tss_1 \doteq (\{\delta, a._-, \_ + \_\}, \{1\}, \{\textbf{(tc0)}, \textbf{(ta)}, \textbf{(d)}\})$ and $tss_2 \doteq (\{\_ + \_\}, \{1\}, \{\textbf{(tc1)}, \textbf{(tc2)}\})$.

It follows from Definition 13 that tss_1 is source preserving since:

1. the conclusions of **(ta)** and **(d)** are unifiable using any substitution, hence using the unifiers of the empty set of premises,
2. and the conclusion of **(tc0)** is unifiable using the unifiers of the premises, i.e., those that evaluate x and x' to the same term and y and y' to the same term.

It then follows from Definition 14 that $tss_1 \cup tss_2$ grants time transitions over *MPA* terms since

1. tss_1 is strictly stratified using a simple measure of size on terms, it is source preserving as shown before, and by applying unifiers to the source of conclusion of **(tc0)**, **(ta)** and **(d)**, i.e., the set $\{x + y, a.x, \delta\}$, we can cover the syntax of *MPA*,
2. in tss_2, deduction rules **(tc1)** and **(tc2)** have source-dependent negative premises with label 1 (note that **(tc1)** and **(tc2)** are the same as their reduced versions).

From Theorem 3, it follows that the extension of tss_m with $tss_1 \cup tss_2$ is a granting extension, hence an orthogonal extension. Furthermore, the extension of $tss_m \cup tss_1 \cup tss_2$ with tss_0 is conservative, hence orthogonal, following Theorem 1. Since orthogonality is a preorder, we conclude that $tss_m \cup tss_t$ is an orthogonal extension of tss_m.

The following theorem establishes the link between orthogonality and equational conservativity. It is very similar to the theorem stated in [16, 17] about the relation between operational and equational conservativity. The theorem states that a sound axiomatization of an operationally conservative extension cannot induce new equalities on the old syntax.

Theorem 4 (Equational Conservativity Theorem) Consider TSS's $tss_0 = (\Sigma_0, L_0, D_0)$ and $tss_1 = (\Sigma_1, L_1, D_1)$ where tss_1 is an orthogonal extension of tss_0. Also let E_0 be a sound and complete axiomatization of tss_0 and E_1 be a sound axiomatization of tss_1. If $\forall_{p,p' \in C(\Sigma_0)}\ E_0 \vdash p = p' \Rightarrow E_1 \vdash p = p'$ then E_1 is an equational conservative extension of E_0.

Finally, the last theorem establishes sufficient conditions for a sound equationally conservative extension to be a complete equational theory for the extended language.

Theorem 5 (Elimination Theorem) Consider TSS's $tss_0 = (\Sigma_0, L_0, D_0)$ and $tss_1 = (\Sigma_1, L_1, D_1)$ where tss_1 is an orthogonal extension of tss_0. Also let E_0 and E_1 be sound axiomatizations of tss_0 and tss_1, respectively. If E_0 is also complete for tss_0, E_1 is an equational conservative extension of E_0 and E_1 eliminates terms from $\Sigma_1 \setminus \Sigma_0$, then E_1 is complete for tss_1.

A typical line of reasoning starts with taking an orthogonal extension and a sound axiomatization thereof, and proving equational conservativity using Theorem 4. Then, by proving an elimination result for the newly introduced operators, one can get completeness of the axiomatization following Theorem 5.

6 Conclusions

In this paper, we defined a more relaxed notion of operational conservativity, called orthogonality which allows for non-destructive extension of the behavior

of the old language. We gave a meta-theorem providing sufficient conditions for this notion. Also, we presented a slightly more general notion of equational conservativity and established the link between these two notions.

Extending the theory presented in this paper with the concept of variable binding is an straightforward extension along the lines of [5]. The second enhancement of our work concerns operational extensions that require a translation of labels (using a kind of abstraction function). Finally, investigating the possibility of other realizations of orthogonality is an interesting subject for future research.

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. Embedding untimed into timed timed process algebra: the case for explicit termination. *MSCS*, 13(4):589–618, 2003.
3. J. C. M. Baeten and C. Verhoef. Concrete Process Algebra. In *Handbook of Logic in Computer Science*, volume 4, pages 149–268. Oxford University Press, 1995.
4. R. Bol and J. F. Groote. The meaning of negative premises in transition system specifications. *JACM*, 43(5):863–914, 1996.
5. W. J. Fokkink and C. Verhoef. A conservative look at operational semantics with variable binding. *I&C*, 146(1):24–54, 1998.
6. R. J. van Glabbeek. The meaning of negative premises in transition system specifications II. *JLAP*, 60-61:229–258, 2004.
7. R.J. van Glabbeek. The linear time - branching time spectrum I. In *Handbook of Process Algebra, Chapter 1*, pages 3–100. Elsevier, 2001.
8. J. F. Groote. Transition system specifications with negative premises. *TCS*, 118(2):263–299, 1993.
9. G. Leduc and L. Leonard. A timed LOTOS supporting a dense time domain and including new timed operators. In *Proceedings of FORTE'92*, pages 87–102. North-Holland, 1993.
10. C.A. Middelburg. An alternative formulation of operational conservativity with binding terms. *JLAP*, 55(1/2):1–19, 2003.
11. M.R. Mousavi and M. A. Reniers. Orthogonal Extensions in Structural Operational Semantics. Technical Report, Dept. of Computer Science, Eindhoven Univ. of Tech., 2005.
12. D. M. Park. Concurrency and automata on infinite sequences. In *Proceedings of the 5th GI Conference*, volume 104 of *LNCS*, pages 167–183. Springer, 1981.
13. G. D. Plotkin. A structural approach to operational semantics. *JLAP*, 60:17–139, 2004.
14. J. J. Vereijken. *Discrete Time Process Algebra*. PhD thesis, Department of Computer Science, Eindhoven University of Technology, 1997.
15. C. Verhoef. A general conservative extension theorem in process algebra. In *Proceedings of PROCOMET'94*, pages 274–302. Elsevier, 1994.
16. C. Verhoef. A congruence theorem for structured operational semantics with predicates and negative premises. *Nordic Journal of Computing*, 2(2):274–302, 1995.
17. C. Verhoef, L. Aceto, and W. Fokkink. Conservative extension in structural operational semantics. *BEATCS*, 69:110–132, 1999.