A general conservative extension theorem in process algebras with inequalities

Pedro R. D'Argenio[1*] and Chris Verhoef[2]

[1]*Dept. of Computer Science. University of Twente.*
P.O.Box 217. 7500 AE Enschede. The Netherlands.
dargenio@cs.utwente.nl

[2]*Faculty of Mathematics and Computer Science. University of Amsterdam*
Kruislaan 403. 1098 SJ Amsterdam. The Netherlands.
chris@fwi.uva.nl

June 1996

Abstract

We prove a general conservative extension theorem for transition system based process theories with easy-to-check and reasonable conditions. The core of this result is another general theorem which gives sufficient conditions for a system of operational rules and an extension of it in order to ensure conservativity, that is, provable transitions from an original term in the extension are the same as in the original system. As a simple corollary of the conservative extension theorem we prove a completeness theorem. We also prove a general theorem giving sufficient conditions to reduce the question of ground confluence modulo some equations for a large term rewriting system associated with an equational process theory to a small term rewriting system under the condition that the large system is a conservative extension of the small one. We provide many applications to show that our results are useful. The applications include (but are not limited to) various real and discrete time settings in ACP, ATP, and CCS and the notions projection, renaming, state operator, priority, recursion, the silent step, autonomous actions, the empty process, divergence, etc.

1991 Mathematics Subject Classification: 68Q05, 68Q10, 68Q42, 68Q55, 03C05.
1991 CR Categories: D.3.1, F.1.1, F.3.2, F.4.3.
Keywords: structured operational semantics, term deduction system, transition system specification, semantic equivalence, semantic preorder, algebraic system, process algebra, conservative extension.

*Supported by the NWO/SION project 612-33-006.

1 Introduction

In the past few years people working in the area of process algebra have started to extend process theories such as CCS, CSP, and ACP with, for instance, real-time or probabilistics. A natural question is whether or not such an extension is somehow related with its subtheory, for instance, whether or not the extension is conservative in some sense. If we add new operators or rules to a particular transition system it would be nice to know whether or not provable transitions of a term in the original system are the same as those in the extended system for that term; we will call this property *operational conservativity* (cf. [GV92]). Or, if we extend an axiomatical framework with new operators, equations, or inequalities it would be interesting to know whether or not a theorem (for instance, an equality or an inequality) in the extended framework over original closed terms can also be derived in the original framework. When no new theorems over closed terms in the original framework are provable from the extension, we call the extension an *algebraic conservative* extension. This is a well-known property under the name of conservativity; we just added the adjective 'algebraic' to prevent possible confusion with the operational variant. In particular we say *equational* or *inequational* conservative extension when the involved algebraic frameworks are, respectively, equational or inequational specifications.

A frequently used method to prove that an algebraic theory is a conservative extension of a subtheory is term rewriting analysis. In process algebra such an analysis is often very complex because the rewriting system associated with a process algebra seems to need term rewriting techniques modulo the equations without a clear direction (such as commutativity of the choice). Moreover, these term rewriting systems generally have undesirable properties making a term rewriting analysis a complex tool for conservativity. Such term rewriting systems are not regular, which implies that confluence (modulo some equations) is not straightforward and we note that the term rewriting relation induced by the rewrite rules does not necessarily commute with the equality induced by the algebraic system, which means that termination modulo these equations is not at all easy to prove. Let us briefly mention two examples to make the problems a bit more concrete. Bergstra and Klop [BK84] claim that for the confluence modulo some equations of their term rewriting system, they need to check ± 400 cases (which they left to the reader as an exercise). Jouannaud communicated to us that, in general, it is very hard (and unreliable) to make such exercises by hand but they can possibly be checked by computer. Our second example originates from Akkerman and Baeten [AB90]. They show that a fragment of ACP with the branching τ is both terminating and confluent modulo associativity and commutativity of the alternative composition. Akkerman told us that it is not clear to him how this result could also be established for the whole system and thus yielding a conservativity result. However, according to Baeten it is not a problem to establish these results; needless to say that their term rewriting analysis is rather complicated.

To bypass the abovementioned problems involving term rewriting, we propose an alternative method to prove conservativity. We provide a general theorem with reasonable and easy-to-check conditions giving us immediately the operational and algebraic conservativity in many cases. For instance, with our results, the conservativity of the abovementioned systems with problematic term rewriting properties is peanuts. The idea is that we transfer the question of algebraic conservativity to that of operational conservativity rather than to perform a term rewriting analysis. The only thing that remains to be done in order to prove the operational conservativity is to check our simple conditions for the operational rules. For the algebraic conservativity we moreover demand completeness of the subtheory and soundness of its extension. These conditions are in our opinion reasonable, because relations between algebraic theories only become important if the theories themselves satisfy such well-established basic requirements. Moreover, our result works for a large class of theories, which is certainly not the case with a term rewriting analysis. All this implies that we give a semantical proof of conservativity, which might be seen as a drawback since a term rewriting analysis often is model independent (but see Bergstra and Klop [BK85], Fokkink and Zantema [FZ94], Zantema [Zan95] for semantical term rewriting analyses). However, since the paper of Plotkin [Plo81], the use of labelled transition systems as a model for operational semantics of process theories is widespread; so virtually every process theory has an operational semantics of this kind. Moreover, our algebraic conservativity result holds for all semantical preorders –thus, also equivalences– that are definable exclusively in terms of transition relations. We recall some examples of semantical preorders and equivalences which are definable in terms of relation and predicate symbols only to show that our conditions are quite general. Examples of equivalences are trace equivalence, failure equivalence, simulation equivalence, strong bisimulation equivalence, weak bisimulation equivalence, branching bisimulation equivalence, the rooted variants of the last two equivalences, etc. We refer to van Glabbeek's linear time – branching time spectra [Gla90] and [Gla93] for more information on these equivalences. In [Gla90] and [Gla93], references to the origins (and use) of these semantics can be found. Equivalences for true concurrency were also defined in that way, for instance, step bisimulation [NT84, BB93] and pomset bisimulation [BC88]. Examples of preorders are simulation, n-nested simulations [GV92], ready simulation [BIM95], the preorder for the degree of parallelism based on pomset bisimulation of [Ace91], the "more distributed than" preorders of [Cas93] and [Yan93], the preorder for unstable nondeterminism of [VB96] and the preorders of bisimulation with divergence of [Abr87] and [Wal90].

As a result we now can prove conservativity without using the confluence property. However, it is widely recognised that confluence itself is an important property, for instance, for computational or implementational purposes. So, at this point the question arises: "Why bother about such a general conservative extension theorem if we still have to prove confluence for each particular system and get the conservativity as a by-product?" The answer is that once we have the conservativity we can considerably reduce the complexity of the ground confluence as a by-product. We prove a general reduction theorem stating that in many cases a conservative extension is ground Church-Rosser modulo some equations if the basic system already has this property.

For instance, the 400 cases of Bergstra and Klop [BK84] reduce to a term rewriting analysis with only five rewrite rules and two equations. We should note, however, that they prove (modulo 400 cases!) the confluence for open terms (although they only need the closed case), whereas our reduction theorem gives the closed case only. In fact, we show that conservativity and ground Church-Rosser are, in some sense, equally expressive properties.

Another advantage of our approach is that it also works for process algebras with really bad term rewriting properties, such as process algebras containing the three τ laws of Milner, where the term rewriting approach breaks down; see, e.g., [BK85]. We will treat these examples in this paper.

Now that we have given some motivation for this paper we discuss its organisation. In section 2 we recall some general SOS definitions of Verhoef [Ver95] and in section 3 we recall some concepts of algebraic systems. We provide a running example to elucidate the abstract notions. In section 4 we formally define the notions of operational and algebraic conservativity. Then we prove a general operational conservativity theorem, a general inequational conservativity theorem and a simple corollary concerning completeness. Also here we provide our running example. In the next section we recall some basic term rewriting terminology to prove the abovementioned reduction theorem on the ground Church-Rosser property modulo some equations. In section 6 we give the reader an idea of the applicability of our general theorems. Surprisingly, we could not find any conservativity result in the literature for which our conservativity theorem could not be applied, as well. The last section contains concluding remarks.

1.1 Related work

In this subsection we briefly mention related work. Nicollin and Sifakis [NS91] prove conservativity—in some particular cases—using the same general approach as we propose in this paper, namely a semantical approach. We will discuss their conservativity results (and new results) in section 6. The notion that we call in this paper operational conservativity originates from Groote and Vaandrager [GV92] under the name conservativity. In Groote [Gro93], Bol and Groote [BG91], and Fokkink and Verhoef [FV95] this notion also appears. In all these papers this notion is used for a different purpose than ours. Aceto, Bloom and Vaandrager [ABV92] introduce a so-called disjoint extension, which is a more restricted form of an operational conservative extension; they need this restriction for technical reasons. They present an algorithm generating a sound and complete axiomatisation if the operational rules satisfy certain criteria. Bosscher [Bos94] studied term rewriting properties of such axiomatisations by looking at the form of the operational rules.

2 Some general SOS definitions

In this section we briefly recall some notions concerning general SOS theory that we will need later on in Section 4. We follow Verhoef [Ver95] since this paper gives the most general setting. To elucidate the formal notions we intersperse them with a running example.

We assume that we have an infinite set V of variables with typical elements $x, y, z, \ldots$. A *(single sorted) signature* Σ is a set of function symbols together with their arity. If the arity of a function symbol $f \in \Sigma$ is zero we say that f is a *constant symbol*. The notion of a *term* (over Σ) is defined as usual: $x \in V$ is a term; if $t_1, \ldots, t_n$ are terms and if $f \in \Sigma$ is n-ary then $f(t_1, \ldots, t_n)$ is a term. A term is also called an *open* term; if it contains no variables we call it *closed*. We denote the set of closed terms by $C(\Sigma)$ and the set of (open) terms by $O(\Sigma)$. We also want to speak about variables occurring in terms: let $t \in O(\Sigma)$ then $var(t) \subseteq V$ is the set of variables occurring in t.

A *substitution* σ is a map from the set of variables into the set of terms over a given signature. This map can easily be extended to the set of all terms by substituting for each variable occurring in an open term its σ-image.

Definition 2.1 A *term deduction system* is a structure (Σ, D) with Σ a signature and D a set of *deduction rules*. The set $D = D(T_p, T_r)$ is parameterised with two sets, which are called (following usual process algebra terminology) respectively the set of *predicate symbols* and the set of *relation symbols*. Let $s, t, u \in O(\Sigma)$, $P \in T_p$, and $R \in T_r$. We call expressions $Ps, \neg Ps$, tRu, and $t\neg R$ *formulas*. We call the formulas Ps and tRu *positive* and $\neg Ps$ and $t\neg R$ *negative*[1]. If S is a set of formulas we write $PF(S)$ for the subset of positive formulas of S and $NF(S)$ for the subset of negative formulas of S.

A deduction rule $d \in D$ has the form

$$\frac{H}{C}$$

with H a set of formulas and C a positive formula; we will also use the notation H/C. We call the elements of H the *hypotheses* of d and we call the formula C the *conclusion* of d. If the set of hypotheses of a deduction rule is empty we call such a rule an *axiom*. We denote an axiom simply by its conclusion provided that no confusion can arise. The notions "substitution", "var", and "closed" extend to formulas and deduction rules as expected. Note that the overload of the symbol C in $C(\Sigma)$ and H/C is harmless.

Let $d = H/C$ a deduction rule with $C = Ps$ or $C = sRs'$. Let $X = var(s)$ and let $Y = \bigcup\{var(t') |\, tRt' \in H\}$. If $var(d) = X \cup Y$ we call d *pure*. A term deduction system is called *pure* if all its rules are pure. $\Box$

Note that arbitrary many premises are allowed in the set of hypotheses of a deduction rule. This generality is useful, for instance, in real-time process algebras where it is very natural to have continuously many premises (see [MT90, Yi90, Klu93].)

[1]The idea behind $t\neg R$ is that there is no term s such that tRs. We chose this notation among others like $\neg tR$, $\neg Rt$, or $\neg(tR)$ since it seems to be the most accurate one.

Example 2.2 As a running example, we present the operational semantics of the process algebra with parallel composition PA [BK84, BW90] and the basic process algebra with relative discrete time: $\mathrm{BPA_{dt}}$ [BB92]. We will consider separately BPA (basic process algebra), MRG, a module that defines parallel processes without communication and DT, which is an extension to discrete timed processes.

The signature of BPA contains constants a of a set $\mathbf{A}$ of *atomic actions*, *alternative composition*, denoted $+$, and *sequential composition* $(\cdot)$. The signature of MRG (for merge) contains *parallel composition* or *merge* $(||)$ and the *left merge* $(\lfloor\!\lfloor)$. The signature of DT contains $+$, $\cdot$ and the *discrete time unit delay* (σ_d).

It is easy to see that the signatures of BPA, MRG, and DT with their operational rules in Table 1 form term deduction systems. These term deduction systems have relations $\_\xrightarrow{a}\_$ for all $a \in \mathbf{A}$, a relation $\_\xrightarrow{\sigma}\_$ with $\sigma \notin \mathbf{A}$ and predicates $\_\xrightarrow{a}\surd$ for all $a \in \mathbf{A}$. The intended interpretation of $x \xrightarrow{a} x'$ is that a process x may execute an action a and evolve into x'. The meaning of $x \xrightarrow{a} \surd$ is that x terminates successfully after the execution of a. With $x \xrightarrow{\sigma} x'$ we mean that a process x evolves into x' by letting a time unit pass. We write $x \not\xrightarrow{\sigma}$ instead of $x \neg \xrightarrow{\sigma}$.

Our running examples are the combination of BPA with either MRG or DT. The signature of the term deduction system PA is the union of signature of BPA and MRG. The operational rules are those of BPA and MRG combined. Similarly, the term deduction system $\mathrm{BPA_{dt}}$ is obtained by combining BPA and DT. It is easy to check that PA and $\mathrm{BPA_{dt}}$ are indeed term deduction systems. We will later on use them to demonstrate our results. □

(i)

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

$$\frac{x \xrightarrow{a} x'}{\begin{array}{c} x + y \xrightarrow{a} x' \\ y + x \xrightarrow{a} x' \end{array}} \qquad \frac{x \xrightarrow{a} \surd}{\begin{array}{c} x + y \xrightarrow{a} \surd \\ y + x \xrightarrow{a} \surd \end{array}}$$

(ii)

$$\frac{x \xrightarrow{a} x'}{\begin{array}{c} x||y \xrightarrow{a} x'||y \\ y||x \xrightarrow{a} y||x' \end{array}} \qquad \frac{x \xrightarrow{a} x'}{x \lfloor\!\lfloor y \xrightarrow{a} x'||y}$$

$$\frac{x \xrightarrow{a} \surd}{\begin{array}{c} x||y \xrightarrow{a} y \\ y||x \xrightarrow{a} y \end{array}} \qquad \frac{x \xrightarrow{a} \surd}{x \lfloor\!\lfloor y \xrightarrow{a} y}$$

(iii)

$$\sigma_d(x) \xrightarrow{\sigma} x \qquad \frac{x \xrightarrow{\sigma} x' \quad y \xrightarrow{\sigma} y'}{x + y \xrightarrow{\sigma} x' + y'}$$

$$\frac{x \xrightarrow{\sigma} x'}{x \cdot y \xrightarrow{\sigma} x' \cdot y} \qquad \frac{x \xrightarrow{\sigma} x' \quad y \not\xrightarrow{\sigma}}{\begin{array}{c} x + y \xrightarrow{\sigma} x' \\ y + x \xrightarrow{\sigma} x' \end{array}}$$

Table 1: Operational rules for BPA, MRG and DT

Definition 2.3 Let T be a term deduction system. Let $F(T)$ be the set of all closed formulas over T. We denote the set of all positive formulas over T by $PF(T)$ and the negative formulas by $NF(T)$. Let $X \subseteq PF(T)$. We define when a formula $\varphi \in F(T)$ *holds in* X; notation $X \vdash \varphi$.

$$\begin{array}{lll} X \vdash sRt & \text{if} & sRt \in X, \\ X \vdash Ps & \text{if} & Ps \in X, \\ X \vdash s\neg R & \text{if} & \forall t \in C(\Sigma) : sRt \notin X, \\ X \vdash \neg Ps & \text{if} & Ps \notin X. \end{array}$$

□

The purpose of a term deduction system is to define a set of positive formulas that can be deduced using the deduction rules. For instance, if the term deduction system is a transition system specification then a transition relation is such a set. For term deduction systems without negative formulas this set comprises all the formulas that can be proved by a well-founded proof tree. If we allow negative formulas in the premises of a deduction rule it is no longer obvious which set of positive formulas can be deduced using the deduction rules. Bloom, Istrail, and Meyer [BIM88, BIM95] formulate that a transition relation must agree with a transition system specification. We will use their notion; it is only adapted to incorporate predicates.

Definition 2.4 Let $T = (\Sigma, D)$ be a term deduction system and let $X \subseteq PF(T)$ be a set of positive closed formulas. We say that X *agrees with* T if for every formula $\varphi \in X$ we have that there is a deduction rule instantiated with a closed substitution such that the instantiated conclusion equals φ and all the instantiated hypotheses hold in X, and vice versa. More formally: X agrees with T if

$$\varphi \in X \iff \exists\, H/C \in D, \sigma : V \to C(\Sigma) : \sigma(C) = \varphi, \forall h \in H : X \vdash \sigma(h).$$

□

There are several ways to give meaning to a set of formulas that agree with a given term deduction system. In [Gla95] an elaborate study on the meaning of negative premises is given reviewing known interpretations and discussing new ones. We mention the uniqueness approach of [BIM95], the stratification techniques described in [Gro93], the reduction techniques of [BG91], and the complete models of [Gla95]. In this paper we focus on applications instead of theory so we choose to work with a technique that is easily applicable: the stratification technique described in [Gro93]. We note that our results are also valid for more general models such as stable ones [Gla95]. We refer the interested reader to [FV95] for details.

Definition 2.5 Let $T = (\Sigma, D)$ be a term deduction system. A mapping $S : PF(T) \to \alpha$ for an ordinal α is called a *stratification* for T if for all deduction rules $H/C \in D$ and closed substitutions σ the following conditions hold:

1. for all $h \in PF(H)$, $S(\sigma(h)) \leq S(\sigma(C))$;
2. for all $s\neg R \in NF(H)$ and for all $t \in C(\Sigma)$, $S(\sigma(sRt)) < S(\sigma(C))$;
3. for all $\neg Ps \in NF(H)$, $S(\sigma(Ps)) < S(\sigma(C))$.

We call a term deduction system *stratifiable* if there exists a stratification for it. □

Example 2.6 When dealing with GSOS languages [BIM95], a stratification is obtained just by measuring the complexity of a positive formula in terms of counting a particular symbol occurring in the conclusion of a rule with negative antecedents. This does not hold in general for any term deduction system but can be adopted as a rule of thumb. In our case, for BPA and PA the stratifications are trivial since they have no negative rule. We can see in Table 1 that $\mathrm{BPA}_{\mathrm{dt}}$ has only one rule with a negative antecedent. In its conclusion we find the function symbol $+$. Let t be a closed term with n occurrences of this symbol. Then the map $S(t \xrightarrow{\sigma} t') = n$ is a stratification (t' is a closed term). This means that the term deduction system $\mathrm{BPA}_{\mathrm{dt}}$ makes sense. Informally speaking this means that the transition relations and the predicates are defined by the operational rules. □

Next, we assign to a term deduction system a (regular) ordinal that expresses a uniform upper bound of the number of premises in the deduction rules. We use this upper bound for proof-technical reasons.

Definition 2.7 Let V be a set. If $0 \leq |V| < \aleph_0$ we define the *degree* of V, denoted $d(V)$, to be ω_0. If $|V| = \aleph_\alpha$ for an ordinal $\alpha \geq 0$, we define $d(V) = \omega_{\alpha+1}$.

Let $T = (\Sigma, D)$ be a term deduction system. The degree $d(H/C)$ of a deduction rule $H/C \in D$ is the degree of its set of positive premises: $d(H/C) = d(PF(H))$. Let $\omega_\alpha = \sup\{d(H/C) |\ H/C \in D\}$. The degree $d(T)$ of a term deduction system T is ω_0 if $\alpha = 0$ and $\omega_{\alpha+1}$ otherwise. □

Example 2.8 The reader can see that for every rule H/C in Table 1, $|PF(H)| \leq 2$. Thus, $d(H/C) = \omega_0$, which implies that the degree of every term deduction system in our example is ω_0. In particular, $d(\mathrm{BPA}) = d(\mathrm{PA}) = d(\mathrm{BPA}_{\mathrm{dt}}) = \omega_0$. □

Next, we will define a set of positive formulas from which we will show that it agrees with a given term deduction system.

Definition 2.9 Let $T = (\Sigma, D)$ be a term deduction system and let $S : PF(T) \to \alpha$ be a stratification for an ordinal number α. We define a set $T_S \subseteq PF(T)$ as follows.

$$T_S = \bigcup_{i<\alpha} T_i, \quad T_i = \bigcup_{j<d(T)} T_{i,j}.$$

We will need unions over T_i and $T_{i,j}$ in proofs; so, we introduce the following notations

$$U_i = \bigcup_{i'<i} T_{i'} \ (i \leq \alpha), \quad U_{i,j} = \bigcup_{j'<j} T_{i,j'} \ (j \leq d(T)).$$

Now we define for all $i < \alpha$ and for all $j < d(T)$ the set $T_{i,j}$:

$$\begin{aligned} T_{i,j} \;=\; & \Big\{\varphi \mid S(\varphi) = i,\ \exists H/C \in D \text{ and } \sigma : V \to C(\Sigma) \text{ with } \sigma(C) = \varphi, \\ & \quad \forall\, h \in \mathit{PF}(H) : U_{i,j} \cup U_i \vdash \sigma(h) \text{ and } \forall\, h \in \mathit{NF}(H) : U_i \vdash \sigma(h)\Big\}. \end{aligned}$$

□

Example 2.10 We will elucidate the above definition by calculating a specific set T_S. The example is taken from [Ver95] and is based on an example of [Gro93]. Consider the term deduction system T with only a constant c in the signature, and rules $\neg P_n c/P_{n+2}c$ and $\neg P_{2n}c/P_0 c$ with $n \geq 0$. Then $S : \mathit{PF}(T) \to 2\omega_0$ defined as $S(P_{2n}c) = \omega + n$ and $S(P_{2n+1}c) = n$ is a stratification for T. Moreover $d(T) = \omega_0$. Now, we calculate T_S. Since there are no positive premises we have that $T_{i,0} = T_{i,j}$ for all $j < d(T)$. So $T_i = T_{i,0}$. It is not hard to verify that for all $n \geq 0$ we have $T_{2n} = T_{\omega_0+2n+1} = \emptyset$, $T_{2n+1} = \{P_{4n+3}c\}$, and $T_{\omega_0+2n} = \{P_{4n}c\}$. So we find that $T_S = \{P_0c, P_3c, P_4c, P_7c, P_8c, P_{11}c, \ldots\}$. □

The next theorem is taken from Verhoef [Ver95] but its proof is essentially the same as a similar theorem due to Groote [Gro93].

Theorem 2.11 *Let $T = (\Sigma, D)$ be a term deduction system and let $S : \mathit{PF}(T) \to \alpha$ be a stratification for an ordinal number α. Then T_S agrees with T. If S' is also a stratification for T then $T_S = T_{S'}$. That is, every stratifiable term deduction system has a unique set of formulas obtained as in Definition 2.9 that agrees with it.*

Example 2.12 Since the term deduction systems of our running example are stratifiable it follows from the above theorem that the rules of BPA, PA, and BPA_{dt} determine a transition relation (with predicates) on closed terms. □

Definition 2.13 Let $T = (\Sigma, D)$ be a term deduction system and let F be a set of formulas. The *variable dependency graph* of F is a directed graph with variables occurring in F as its nodes. The edge $x \to y$ is an edge of the variable dependency graph if and only if there is a positive relation $tRs \in F$ with $x \in \mathit{var}(t)$ and $y \in \mathit{var}(s)$.

The set F is called *well-founded* if every backward chain of edges in its variable dependency graph is finite. A deduction rule is called well-founded if its set of hypotheses is so. A term deduction system is called well-founded if all its deduction rules are well-founded. □

Example 2.14 It is easy to see that the rules of our running examples are well-founded. □

3 Some concepts of algebraic systems

We want to formulate a general theorem in which both equational specifications and inequational specifications play a crucial role. For completeness sake we will, therefore, recall the necessary notions in this section. We mainly follow [Gil76]; an alternative approach can be found in [Wec92, Hen88].

Next, we define the notion of an algebraic system, or abstract algebra. It will turn out that both equational and inequational systems are special cases of an algebraic system.

Definition 3.1 An *algebraic system* (or *abstract algebra*) is a structure $(\Sigma, \mathcal{A}, \mathcal{P})$ where Σ is a signature, $\mathcal{P} \subseteq O(\Sigma) \times O(\Sigma)$ is a set of predicates, and $\mathcal{A}$ is a set of *axioms* having the form $\{p_i(s_i, t_i) | i \in I\} \Rightarrow p(s, t)$ where I is a finite set, $s, t, s_i, t_i \in O(\Sigma)$ and $p, p_i \in \mathcal{P}$. We call the set $\{p_i(s_i, t_i) | i \in I\}$ the conditions. If the set of conditions is empty we write $p(s, t)$ instead of $\emptyset \Rightarrow p(s, t)$. Note that the overload of the word predicate with that of definition 2.1 is harmless. $\Box$

The predicate symbols in algebraic systems are most often relations such as equality or, in our case, inequality. Next, we axiomatise the most common properties of such predicates.

Definition 3.2 Let $(\Sigma, \mathcal{A}, \mathcal{P})$ where Σ be an algebraic system. Let $p \in \mathcal{P}$, $f \in \Sigma$ be n-ary, and $x, y, z, x_i, y_i \in V$ then,

- if $p(x, x) \in \mathcal{A}$ we say that p is *reflexive*;
- if $\{p(x, y)\} \Rightarrow p(y, x) \in \mathcal{A}$ we say that p is *symmetric*;
- if $\{p(x, y), p(y, z)\} \Rightarrow p(x, z) \in \mathcal{A}$ we say that p is *transitive*;
- if p is both reflexive, symmetric, and transitive, we say that p is an equivalence; and
- if $\{p(x_1, y_1), \ldots, p(x_n, y_n)\} \Rightarrow p(f(x_1, \ldots, x_n), f(y_1, \ldots, y_n)) \in \mathcal{A}$ we say that p *preserves* f.

We refer to the first axiom as the *axiom of reflexivity* and to the others likewise. We refer to some or all of the above axioms loosely as the *special axioms*. $\Box$

Now we are able to give precise definitions of equational and inequational specifications.

Definition 3.3 An *inequational specification* is an algebraic system with a single predicate that is reflexive, transitive and preserves all functions in the signature. An *equational specification* is an inequational specification such that its predicate is also symmetric. $\Box$

From now on we will tacitly assume the presence of the axioms of reflexivity, transitivity, and preservation of functions in the inequational specifications, and in addition symmetry in the equational specifications that we will discuss in the examples and applications.

Example 3.4 Now we give a few examples of equational specifications and inequational ones. We present some axioms and inequalities that fit our running example in a natural way. We begin with the equational specification called BPA. Its signature is the same as the signature of the term deduction system also called BPA. Its axioms are listed in Table 2.(i). This is a known system; see [BW90] for its use.

The equational specification $\mathrm{BPA}_{\mathrm{dt}}$ is constructed in the same way: take the signature of its term deduction system and its axioms are the ones of the equational specification BPA and the ones listed in Table 2.(iv). Also this equational specification is known, see [BB92]. To demonstrate our general theorems we will also need the equational specification DT formed by the same signature as its term deduction system, and axioms in Table 2.(iv). In fact, $\mathrm{BPA}_{\mathrm{dt}}$ is the sum of the modules BPA and DT.

Now we give an example of an inequational specification. The inequational specification $\mathrm{PA}^{\leq}$ consists of the signature of the term deduction system PA together with the axioms in Table 2.(i–iii). In this case, expressions having the form $s = t$ stand for the two axioms $s \leq t$ and $t \leq s$. From now on, we will assume $s = t$ as an abbreviation for those two inequalities in an inequational specification. Thus, for instance, the expression A3 stands for the two axioms $x + x \leq x$ and $x \leq x + x$.

Similar to $\mathrm{BPA}_{\mathrm{dt}}$ we will define two modules that, when combined, form $\mathrm{PA}^{\leq}$. We will use those two inequational specifications to show our main results for inequational systems. The first one called $\mathrm{BPA}^{\leq}$ has the signature of the equational specification BPA, and as axioms those in Table 2.(i–ii) (note that there are eleven axioms). The inequational specification expressing the parallel side of $\mathrm{PA}^{\leq}$ is called $\mathrm{MRG}^{\leq}$ and has the same signature than $\mathrm{PA}^{\leq}$ and the nine axioms in Table 2.(iii).

Now that we have given examples of definition 3.3 we will briefly discuss their axioms. Axioms A1–5 and M1–4 are the well known axioms for the PA process algebra [BK84, BW90]; PA is a simple language with sequential, alternative and parallel composition. Axioms DT1 and DT4 originate from [BB92]; $\mathrm{BPA}_{\mathrm{dt}}$ is a sequential basic language that incorporates discrete time features. See also [BV95] for a systematic treatment of the above equational specifications. We discuss the inequational axioms. SM stands for simulation; to the best of our knowledge this axiom is introduced here. MP stands for "more parallel" and embodies the idea that $x||y$ has a "more parallel behaviour" than $x \cdot y$. Note that for closed terms, MP can be derived with induction on the size of x from the other axioms in $\mathrm{PA}^{\leq}$. □

Noteworthy perhaps is that in some treatments of equational theories, the notion of equational specification does not incorporate any defined behaviour of the equality predicate, but its suggestive name (see, for instance, [Wec92, Hen88, BW90]) or a tacitly assumed presence of equational logic. The meaning of the equality predicate is often expressed in the notion of derivability. Normally this would not give rise to any

(i)			
A1	$x + y = y + x$	A4	$(x + y) \cdot z = x \cdot z + y \cdot z$
A2	$x + (y + z) = (x + y) + z$	A5	$(x \cdot y) \cdot z = x \cdot (y \cdot z)$
A3	$x + x = x$		
(ii)			
SM	$x \leq x + y$		
(iii)			
M1	$x \| y = x \lfloor\!\lfloor y + y \lfloor\!\lfloor x$	M3	$a \cdot x \lfloor\!\lfloor y = a \cdot (x \| y)$
M2	$a \lfloor\!\lfloor x = a \cdot x$	M4	$(x + y) \lfloor\!\lfloor z = x \lfloor\!\lfloor z + y \lfloor\!\lfloor z$
MP	$x \cdot y \leq x \| y$		
(iv)			
DT1	$\sigma_d(x) + \sigma_d(y) = \sigma_d(x + y)$	DT2	$\sigma_d(x) \cdot y = \sigma_d(x \cdot y)$

Table 2: Axioms for the running examples

problems since mostly the application is only equational specifications. In this paper, such an approach would be very confusing since there would be no distinction between the definition of equational specification and that of inequational specification. To make this distinction apparent we put the special axioms in the definition of (in)equational specification instead of in the definition of derivability. As a result the next definition of derivability only contains the substitutivity property since that one is not algebraically expressible.

Definition 3.5 Let $\mathcal{S} = (\Sigma, \mathcal{A}, \mathcal{P})$ be an algebraic system. Let $s, t \in O(\Sigma)$. A statement $p(s, t)$ *can be derived* from $\mathcal{A}$, notation $\mathcal{A} \vdash p(s, t)$, if there is an axiom in $\mathcal{A}$ such that, together with a given substitution, the premises of the axioms can be derived from $\mathcal{A}$, and the conclusion is $p(s, t)$, that is, let $\sigma : V \to O(\Sigma)$, then

$$\left.\begin{array}{l} \{p_i(s_i, t_i) | i \in I\} \Rightarrow p(s, t) \in \mathcal{A} \\ \text{and } \forall i \in I. \mathcal{A} \vdash p_i(\sigma(s_i), \sigma(t_i)) \end{array}\right\} \implies \mathcal{A} \vdash p(\sigma(s), \sigma(t))$$

We call this property the *substitutivity axiom*. □

In the next definition, we borrow the notion of A-assignments from [Hen88].

Definition 3.6 An *algebra* is a set A of elements, the *carrier*, together with certain functions over A of arity $n \geq 0$.

Let Σ be a signature. A Σ-*algebra* A is an algebra with a function f_{A} for each function symbol $f \in \Sigma$ with the same arity. Such a correspondence is called an *interpretation*. An A-*assignment for* V is a function $\rho : V \to A$. Let $h_\rho : O(\Sigma) \to A$ be the homomorphism defined inductively as follows:

- $h_\rho(f(t_1, \ldots, t_n)) = f_{\mathsf{A}}(h_\rho(t_1), \ldots, h_\rho(t_n))$; and
- $h_\rho(x) = \rho(x)$.

It can be shown that h_ρ is the unique homomorphism from $O(\Sigma)$ to A such that $h_\rho(x) = \rho(x)$ [Hen88].

Let $\mathcal{S} = (\Sigma, \mathcal{A}, \mathcal{P})$ be an algebraic system. Let A be a Σ-algebra with carrier set A. Let $\mathcal{R}$ be a set of binary relations on A with one relation p_A for each $p \in \mathcal{P}$. For $s, t \in O(\Sigma)$ we say that $p(s,t)$ *holds in* A *under* $\mathcal{R}$, notation $\mathsf{A}/_{\mathcal{R}} \models p(s,t)$ if for all A-assignment ρ for V, we have $p_A(h_\rho(s), h_\rho(t))$.

$\mathcal{A}$ is a *sound axiomatisation with respect to* $\mathcal{R}$ *for* A if for all $s, t \in O(\Sigma)$, $p \in \mathcal{P}$

$$\mathcal{A} \vdash p(s,t) \implies \mathsf{A}/_{\mathcal{R}} \models p(s,t).$$

Moreover, if for all closed terms $s, t \in C(\Sigma)$ and $p \in \mathcal{P}$

$$\mathcal{A} \vdash p(s,t) \iff \mathsf{A}/_{\mathcal{R}} \models p(s,t)$$

then $\mathcal{A}$ is called a *complete axiomatisation with respect to* $\mathcal{R}$ *for* A. □

A model for our examples

Now, we will briefly discuss the semantics of our running examples and give the necessary definitions. We state this in a separate paragraph since some new results are introduced. We will use them later on to demonstrate our main theorems. First we give the definition of simulation and that of bisimulation [Par81], adapted to the running examples.

Definition 3.7 A binary relation S on the set of closed PA terms is a *simulation* if for all $(s,t) \in S$ and for all $a \in \mathbf{A}$, the two following *transfer properties* hold:

- $\forall s' : s \xrightarrow{a} s' \implies \exists t' : t \xrightarrow{a} t'$ and $(s', t') \in S$,
- $s \xrightarrow{a} \surd \implies t \xrightarrow{a} \surd$.

If there is a simulation S such that $(s,t) \in S$, then s *is simulated by* t, notation $s \underrightarrow{\subseteq} t$.

Now we give the well-known definition of bisimulation modified to our case. First, extend the notion of simulation by considering also the relation $\xrightarrow{\sigma}$. A binary relation S on the set of closed $\mathrm{BPA_{dt}}$ terms is a *(strong) bisimulation* if S and S^{-1} are simulations. If there is a bisimulation S such that $(s,t) \in S$, then s and t are *bisimilar*, notation $s \underline{\leftrightarrow} t$. □

The facts that BPA is sound and complete modulo strong bisimulation and that $\mathrm{BPA_{dt}}$ is sound with respect to strong bisimulation equivalence are well-known. We refer to [BW90, BV95, BB92] for details.

Since the inequational specifications are new here we will focus more on those.

Lemma 3.8 *The inequational specifications* $\mathrm{BPA}^{\leq}$ *and* $\mathrm{PA}^{\leq}$ *are sound axiomatisations with respect to the* $\subsetsim$ *model induced by their term deduction systems.*

The inequational specification $\mathrm{BPA}^{\leq}$ *is complete with respect to the* $\subsetsim$ *model induced by the* BPA *term deduction system.*

Proof. (Sketch) In order to prove that $\mathrm{PA}^{\leq}$ is a sound axiomatisation with respect to the $\subsetsim$ model induced by the PA term deduction system, it is enough to prove that $\subsetsim$ is reflexive, transitive, and preserves all functions in PA (i.e. $\subsetsim$ is a precongruence for PA) and moreover, that for every axiom $s \leq t$ of $\mathrm{PA}^{\leq}$ in Table 2 with free variables in V, the relation

$$S = \{(\sigma(s), \sigma(t)) |\ \sigma \text{ substitutes closed terms for variables in } V\} \cup Id$$

is a simulation. As a consequence, $\mathrm{BPA}^{\leq}$ is also a sound axiomatisation with respect to $\subsetsim$.

Moreover, $\mathrm{BPA}^{\leq}$ is a complete axiomatisation with respect to the $\subsetsim$ model induced by the BPA term deduction system. The proof follows by induction on the size of the basic terms [BW90] by considering that if t is a basic term then $t \xrightarrow{a} t'$ (respectively $t \xrightarrow{a} \surd$) if and only if t has the form $t'' + (a \cdot t')$ (respectively $t'' + a$) modulo axioms A1, A2. □

The equational specification $\mathrm{PA}^{\leq}$ is also complete as we will show using our results later on.

4 Operational and algebraic conservativity

In this section we prove a general operational conservative extension theorem with easy to check conditions. We also study conservativity on algebraic systems and we state that algebraic conservative extension can be derived from conservativity on models which are complete for the original algebraic system. If we moreover have the elimination property for the new operators we also have completeness of the extension. By combining both results, we prove as a corollary a general inequational conservative extension theorem. We will use our running examples to elucidate the definitions and to demonstrate our results. We recall that since an equational specification is a special case of an inequational specification, all our results also hold for equational specifications, which is indeed a very important subcase.

Definition 4.1 Let Σ_0 and Σ_1 be signatures. If for all $f \in \Sigma_0 \cap \Sigma_1$ the arity of f in Σ_0 is the same as the arity of f in Σ_1 then $\Sigma_0 \oplus \Sigma_1$, called the *sum* of Σ_0 and Σ_1, is the signature $\Sigma_0 \cup \Sigma_1$. Note that $\oplus$ is not simply the union of two signatures since the sum could be undefined if the signatures share a function symbol having different arity for each one of them. $\square$

Example 4.2 We denote by Σ_{BPA} the signature of BPA and similarly for MRG, DT, etc. It is easy for the reader to check that $\Sigma_{\mathrm{BPA}} \oplus \Sigma_{\mathrm{MRG}}$ is defined and is equal to the signature of PA, and that $\Sigma_{\mathrm{BPA}} \oplus \Sigma_{\mathrm{DT}}$ is also defined and equals the signature of $\mathrm{BPA}_{\mathrm{dt}}$. $\square$

Definition 4.3 Let $T^i = (\Sigma_i, D_i)$ be term deduction systems with predicate and relation symbols T_p^i and T_r^i respectively $(i = 0, 1)$. Let $\Sigma_0 \oplus \Sigma_1$ be defined. The sum $T^0 \oplus T^1$, called the sum of T^0 and T^1, is the term deduction system $(\Sigma_0 \oplus \Sigma_1, D_0 \cup D_1)$ with predicate and relation symbols $T_p^0 \cup T_p^1$ and $T_r^0 \cup T_r^1$. $\square$

Example 4.4 Consider the term deduction systems defined in Example 2.2. It is easy to see that $\mathrm{BPA} \oplus \mathrm{MRG} = \mathrm{PA}$ and $\mathrm{BPA} \oplus \mathrm{DT} = \mathrm{BPA}_{\mathrm{dt}}$. $\square$

4.1 Operational conservativity

Next, we formally define the notion of an operational conservative extension and the notion of an operational conservative extension up to some semantical preorder which is defined exclusively in terms of predicate and relation symbols. This is not a serious restriction since many preorders are defined in this way.

The notions operational conservative extension and operational conservative extension up to strong bisimulation equivalence were already defined by Groote and Vaandrager [GV92] (without the adjective 'operational') where they used the first notion to characterise the completed trace congruence induced by their pure well-founded *tyft/tyxt* format. Groote [Gro93] gives the two definitions in the case that negative premises come into play. He used operational conservativity for a similar characterisation result as in [GV92]. In Bol and Groote [BG91] the approach of Groote [Gro93] is placed in a wider perspective. Aceto, Bloom and Vaandrager [ABV92] use a restricted form of operational conservative extension for technical reasons; they call it disjoint extension. Fokkink and Verhoef [FV95] studied conservative extensions in stable term deduction systems with bindings and substitutions. Some corollaries of these results are given in [D'A95] for term deduction systems with unique stable model and terms without bindings and substitutions. We will use the notion of operational conservativity to prove inequational conservativity.

Definition 4.5 Let $T^i = (\Sigma_i, D_i)$ be term deduction systems. Let $T = (\Sigma, D) = T^0 \oplus T^1$ be defined and let $D = D(T_p, T_r)$. The term deduction system T is called an *operational conservative extension* of T^0 if it is stratifiable and for all $s, u \in C(\Sigma_0)$, for all relation symbols $R \in T_r$ and predicate symbols $P \in T_p$, and for all $t \in C(\Sigma)$ we have

$$T_S \vdash sRt \iff T^0_{S^0} \vdash sRt \quad \text{and} \quad T_S \vdash Pu \iff T^0_{S^0} \vdash Pu$$

where S is a stratification for T and S^0 is a stratification for T^0 (take for instance S^0 to be the restriction of S to positive formulas of T^0). $\square$

Definition 4.6 Let $T^i = (\Sigma_i, D_i)$ be term deduction systems with $T = (\Sigma, D) = T^0 \oplus T^1$ defined and let $D = D(T_r, T_p)$. Let ξ be some semantic preorder or equivalence defined in terms of relation and predicate symbols only, i.e., defined in terms of symbols into the set $T_r \cup T_p$. T is an *operational conservative extension of* T_0 *up to* ξ if for all $s, t \in C(\Sigma_0)$, $s \square_\xi^\oplus t \iff s \square_\xi^0 t$, where $\square_\xi^0$ and $\square_\xi^\oplus$ are the preorder or equivalence ξ interpreted in terms of predicate and relation symbols of T^0 and T, respectively.

We will often use $\preceq$ to denote a preorder and $\equiv$ for an equivalence. $\square$

Many preorders and equivalences are definable in terms of relation and predicate symbols only. First we will mention a number of equivalences and then a list of preorders that are defined as such.

Examples of equivalences that satisfy our restrictions are trace equivalence, failure equivalence, simulation equivalence, strong bisimulation equivalence (we recall that we defined this equivalence in 3.7), weak bisimulation equivalence, branching bisimulation equivalence, the rooted variants of the last two equivalences, etc. We refer to van Glabbeek's linear time – branching time spectra [Gla90] and [Gla93] for more information on these equivalences. Equivalences for true concurrency were also defined in that way, for instance, step bisimulation [NT84, BB93] and pomset bisimulation [BC88].

Also many important preorders are defined in terms of relation and predicate symbols. An example of a preorder is simulation that we defined in Definition 3.7. Other examples are n-nested simulations [GV92], ready simulation [BIM95], the preorder for the degree of parallelism based on pomset bisimulation of [Ace91], the "more distributed than" preorders of [Cas93] and [Yan93], the preorder for unstable nondeterminism of [VB96], and the preorders of bisimulation with divergence of [Abr87] and [Wal90].

For all the above equivalences and preorders, the following theorem holds. It states that if an extension is operationally conservative, it is also operationally conservative up to some preorder definable in terms of relations and predicates only.

Theorem 4.7 *Let* $T^i = (\Sigma_i, D_i)$ *be term deduction systems and let* $T = (\Sigma, D) = T^0 \oplus T^1$ *be defined. If* T *is an operational conservative extension of* T^0*, then it is also an operational conservative extension up to* ξ*, for any preorder (thus equivalence)* ξ *defined in terms of predicate and relation symbols only.*

Proof. (Sketch) Let $s, t \in C(\Sigma_0)$. Since T is an operational conservative extension of T^0, the state-transition diagrams (or better: the term-relation-predicate diagrams) of s in both T and T^0 are the same, and so are the term-relation-predicate diagrams of t. Let ξ be a preorder defined in terms of relation and predicate symbols. Because $\preceq_\xi^\oplus$ is defined in the same way for relation and predicate symbols in T^0 as $\preceq_\xi^0$, and the term-relation-predicate diagrams of s and t are the same in both term deduction systems, $s \preceq_\xi^0 t$ implies $s \preceq_\xi^\oplus t$. The counterpositive is proved analogously. $\square$

After publishing [Ver95], Verhoef found out that in his conservativity theorem the *ntyft/ntyxt* format condition was not necessary. Groote was pleasantly surprised by this fact. Shortly thereafter, Verhoef was even more surprised since in [BG91] the conservativity theorem already lacked the *ntyft/ntyxt* format proviso. Groote clarified this mystery by stating that his co-author, Roland Bol, apparently must have seen that the condition was not necessary and had dropped it tacitly. What they *did* discuss in depth [BG91] is their way of giving meaning to negative premises, which is more general than stratifiability. We recall that this paper is focussed towards practical applications. Since the stratification condition is, in our opinion, more practical than their criterion we chose for stratifiability. However, we claim that our theorem can also be proved for that more general notion. In fact, in [D'A95] and [FV95] conservativity theorems have been proved with even more general conditions than the one presented in [BG91]. Anyway, the implications of the fact that the *ntyft/ntyxt* format condition can be dropped are immense. The cross-over between term deduction systems and conditional term rewriting is not longer theoretical [GV92], as can be seen in this paper and in, for instance, [FV95].

After we put the next theorem in context, we discuss the theorem itself. It gives sufficient conditions such that $T^0 \oplus T^1$ is an operational conservative extension of T^0. The theorem is on the one hand a generalisation of a similar result in [BG91], since we allow new rules to contain original function symbols in the left-hand side of a conclusion such as, for instance, rules in Table 1.(*iii*) of our running example. Moreover, Bol and Groote require for the new rules that the left-hand side of a conclusion may not be a single variable, whereas we do not have such a restriction. On the other hand, we use stratifications which is less general than the criterion stated in [BG91].

Theorem 4.8 *Let $T^0 = (\Sigma_0, D_0)$ be a pure well-founded term deduction system. Let $T^1 = (\Sigma_1, D_1)$ be a term deduction system. If there is a conclusion sRt or Ps of a rule $d_1 \in D_1$ with $s \in O(\Sigma_0)$, we additionally require that*

1. *d_1 is pure and well-founded,*
2. *$t \in O(\Sigma_0)$ for premises tRt' of d_1, and*
3. *there is a positive premise containing only Σ_0 terms and a new relation or predicate symbol.*

If $T = T^0 \oplus T^1$ is defined and stratifiable then T is an operational conservative extension of T^0.

Proof. Let $T = (\Sigma, D)$ and $D = D(T_p, T_r)$. Let $S : PF(T) \to \alpha$ be a stratification for T and let $S^0 : PF(T^0) \to \alpha$ be the restriction of S to $PF(T^0)$ (note that S^0 is a stratification).

Let $u, w \in C(\Sigma_0)$, $R \in T_r$, $P \in T_p$, and $v \in C(\Sigma)$. We have to show that the following two bi-implications hold

$$T_S \vdash uRv \iff T^0_{S^0} \vdash uRv,$$
$$T_S \vdash Pw \iff T^0_{S^0} \vdash Pw.$$

By definition 2.9 it suffices to prove the two bi-implications below for all $i < \alpha$.

$$T_i \vdash uRv \iff T_i^0 \vdash uRv, \tag{1}$$
$$T_i \vdash Pw \iff T_i^0 \vdash Pw. \tag{2}$$

We will do this by transfinite induction on i. So let both statements be true for all $i' < i$, then we prove them for i.

We begin to prove both implications from left to right. By definition 2.9 it suffices to show for all $j < d(T)$ that

$$T_{i,j} \vdash uRv \implies T_i^0 \vdash uRv, \tag{3}$$
$$T_{i,j} \vdash Pw \implies T_i^0 \vdash Pw. \tag{4}$$

We will do this by transfinite induction on j. So let 3 and 4 be true for all $j' < j$. We prove them for j. By definition 2.9 there is a rule $d \in D$

$$\frac{\{P_k s_k : k \in K\} \cup \{t_l R_l t'_l : l \in L\} \cup \{\neg P_m u_m : m \in M\} \cup \{v_n \neg R_n : n \in N\}}{C}$$

with $C = sRt$ and a closed substitution σ with $\sigma(s) = u$ and $\sigma(t) = v$. We first show that $d \in D_0$. Suppose that this is not the case. Since $u \in C(\Sigma_0)$ we must have that $s \in O(\Sigma_0)$; so the additional requirements clearly hold for d. Let $var(s) = X$ and $Y = \bigcup_{l \in L} var(t'_l)$. Since d is pure we have that $var(d) = X \cup Y$. We know that $\sigma(x) \in C(\Sigma_0)$ for all $x \in X$. We show that for all $y \in Y$ we have $\sigma(y) \in C(\Sigma_0)$. Suppose that there is a $y_0 \in Y$ with $\sigma(y_0) \in C(\Sigma) \setminus C(\Sigma_0)$ then $\sigma(t'_{l_0}) \in C(\Sigma) \setminus C(\Sigma_0)$. This contradicts the well-foundedness of the rule d, for $U_i \cup U_{i,j} \vdash \sigma(t_{l_0}) R_{l_0} \sigma(t'_{l_0})$ so by the induction hypotheses on i or j we find that $\sigma(t_{l_0}) \in C(\Sigma) \setminus C(\Sigma_0)$. Since t_{l_0} is a Σ_0 term, this must be the result of a substitution. This can only be due to a variable $y_1 \in Y$. By induction on the subsubscript we find an infinite backward chain of edges $y_0 \leftarrow y_1 \leftarrow \ldots$ in the variable dependency graph of d. So $\sigma(y) \in C(\Sigma_0)$ for all $y \in Y$. Let h be a positive premise containing only Σ_0 terms and a new relation or predicate symbol. By definition 2.9 we have $U_i \cup U_{i,j} \vdash \sigma(h)$ so by induction on i or j we find that $U_i^0 \cup U_{i,j}^0 \vdash \sigma(h)$, which is a contradiction since the $\sigma(h)$ is not even a formula in T^0. So the assumption that $d \in D_1$ cannot hold and we must have that $d \in D_0$. This means that d is pure and well-founded. Just as above we can show that $\sigma(x) \in C(\Sigma_0)$ for all $x \in X \cup Y$ so we have that all the instantiated premises of d only contain Σ_0 terms. So we find by induction on i and/or j that for all positive premises h of rule d we have $U_i^0 \cup U_{i,j}^0 \vdash \sigma(h)$. Suppose that $U_i^0 \nvdash \sigma(v_n \neg R_n)$. Then there is a $v'_n \in C(\Sigma_0)$ such that $U_i^0 \vdash \sigma(v_n R_n v'_n)$ so by induction on i we find that also $U_i \vdash \sigma(v_n R_n v'_n)$, which is a contradiction. In this way we find that $U_i^0 \vdash \sigma(h)$ for all negative premises h of rule d. By definition 2.9 we have $T_{i,j}^0 \vdash uRv$ so $T_i^0 \vdash uRv$.

The case $C = Ps$ is treated in the same way. This ends our induction step on j, which proves 3 and 4. So we find that equations 1 and 2 hold from left to right for i.

Now we show that they hold from right to left for i. By definition 2.9 it suffices to show for all $j < d(T^0)$ that

$$T_{i,j}^0 \vdash uRv \implies T_i \vdash uRv,$$
$$T_{i,j}^0 \vdash Pw \implies T_i \vdash Pw.$$

This can be proved by induction on j in the same way as we proved both implications from left to right, but simpler since we can apply induction immediately. This concludes the proof. $\square$

Example 4.9 We are in a position to apply the main results discussed in this section to our running examples.

It is not hard to see that the term deduction systems of BPA and MRG satisfy the conditions of Theorem 4.8. Thus, PA is an operational conservative extension of BPA. Moreover, because of theorem 4.7, PA is an operational conservative extension up to simulation.

Also the term deduction systems of BPA and DT satisfy the conditions of Theorem 4.8; so, $\mathrm{BPA_{dt}}$ is an operational conservative extension of BPA, and again with theorem 4.7 we find that $\mathrm{BPA_{dt}}$ is an operational conservative extension up to strong bisimulation of BPA. $\square$

4.2 Algebraic conservativity

In this subsection we state and prove the main conservativity result for algebraic systems. In particular, we prove that conservativity in inequational and equational specifications with transition system based models is a consequence of operational conservativity. We will use our running examples to show how our results work. For more elaborate application of these results we refer to Section 6.

Definition 4.10 Let $\mathcal{S}_i = (\Sigma_i, \mathcal{A}_i, \mathcal{P}_i)$ be algebraic systems $(i = 0, 1)$. Let $\Sigma_0 \oplus \Sigma_1$ be defined. Then the sum $\mathcal{S}_0 \oplus \mathcal{S}_1$ of $\mathcal{S}_0$ and $\mathcal{S}_1$ is the algebraic system $(\Sigma_0 \oplus \Sigma_1, \mathcal{A}_0 \cup \mathcal{A}_1, \mathcal{P}_0 \cup \mathcal{P}_1)$. $\square$

Example 4.11 Consider the equational and inequational specifications of Example 3.4. Notice that $\mathrm{BPA}^{\leq} \oplus \mathrm{MRG}^{\leq}$ equals $\mathrm{PA}^{\leq}$ and $\mathrm{BPA} \oplus \mathrm{DT}$ equals $\mathrm{BPA_{dt}}$. $\square$

Next, we define the notion of an algebraic conservative extension. An algebraic system is a conservative extension of another one if exactly the same theorems regarding only original terms can be derived from both of them.

Definition 4.12 Let $\mathcal{S}_i = (\Sigma_i, \mathcal{A}_i, \mathcal{P}_i)$ be algebraic systems $(i = 0, 1)$ and let $\mathcal{S} = (\Sigma, \mathcal{A}, \mathcal{P}) = \mathcal{S}_0 \oplus \mathcal{S}_1$ be defined. $\mathcal{S}$ is an *(algebraic) conservative extension* of $\mathcal{S}_0$ if for all $s, t \in C(\Sigma_0)$ and $p \in \mathcal{P}_0$

$$\mathcal{A} \vdash p(s,t) \iff \mathcal{A}_0 \vdash p(s,t).$$

$\square$

In this way, an *inequational conservative extension* is a conservative extension where the involved algebraic systems are inequational specifications, and in particular, an *equational conservative extension* is a conservative extension where the algebraic systems are equational specifications. Notice that in these cases $\mathcal{P}_0 = \mathcal{P}_1$.

Definition 4.13 Let Σ_0 and Σ_1 be two signatures such that $\Sigma_0 \oplus \Sigma_1$ is defined. Let A_0 be a Σ_0-algebra with carrier set A_0. Let A be a $(\Sigma_0 \oplus \Sigma_1)$-algebra with carrier set A such that $A_0 \subseteq A$. A is a *model conservative extension* of A_0 if for every $f \in \Sigma_0$ with arity n, for all $d_1, \ldots, d_n \in A_0$, $f_{\mathsf{A}}(d_1, \ldots, d_n) = f_{\mathsf{A}_0}(d_1, \ldots, d_n)$ where f_{A_0} and f_{A} are the interpretations of f in A_0 and A respectively.

Moreover, let $\mathcal{R}_0$ and $\mathcal{R}$ be sets of binary relations on A_0 and A respectively and let $g : \mathcal{R}_0 \to \mathcal{R}$. $\mathsf{A}/_{\mathcal{R}}$ is a *model conservative extension of* $\mathsf{A}_0/_{\mathcal{R}_0}$ *according to* g if A is a model conservative extension of A_0 and for all $r^0 \in \mathcal{R}_0$, $r^0 = g(r^0) \cap (A_0 \times A_0)$. □

Now, we are in a position to state and prove the main result of this paper. It is the algebraic conservative extension theorem.

Theorem 4.14 *Let $\mathcal{S}_i = (\Sigma_i, \mathcal{A}_i, \mathcal{P}_i)$ be algebraic systems, $i = 0, 1$. Let $\mathcal{S} = (\Sigma, \mathcal{A}, \mathcal{P}) = \mathcal{S}_0 \oplus \mathcal{S}_1$ be defined. Let $\mathcal{A}_0$ be a complete axiomatisation of A_0 with respect to $\mathcal{R}_0$. Let $\mathcal{A}$ be a sound axiomatisation of A with respect to $\mathcal{R}$. If $\mathsf{A}/_{\mathcal{R}}$ is a model conservative extension of $\mathsf{A}_0/_{\mathcal{R}_0}$ according to g such that for all $p \in \mathcal{P}_0$, $g(p_{A_0}) = p_A$, then $\mathcal{S}$ is an algebraic conservative extension of $\mathcal{S}_0$.*

Proof. The proof that for all $s, t \in C(\Sigma_0)$, $\mathcal{A}_0 \vdash p(s,t) \implies \mathcal{A} \vdash p(s,t)$ is trivial. Now, let $s, t \in C(\Sigma_0)$ and suppose $\mathcal{A} \vdash p(s,t)$. Since $\mathcal{A}$ is sound for A respect to $\mathcal{R}$ $\mathsf{A}/_{\mathcal{R}} \models p(s,t)$. Because $\mathsf{A}/_{\mathcal{R}}$ is a model conservative extension of $\mathsf{A}_0/_{\mathcal{R}_0}$ according to g with $g(p_{A_0}) = p_A$, $\mathsf{A}_0/_{\mathcal{R}_0} \models p(s,t)$. Finally, $\mathcal{A}_0 \vdash p(s,t)$ since $\mathcal{A}_0$ is complete for A_0 with respect to $\mathcal{R}_0$. So $\mathcal{A}$ is an algebraic conservative extension of $\mathcal{A}_0$. □

Notice that the requirements on g become trivial in inequational specifications since there is only one predicate to consider (namely, $\leq$). Thus, as a corollary we have the following theorem with many useful applications (see Section 6).

Theorem 4.15 *Let $\mathcal{S}_i = (\Sigma_i, \mathcal{A}_i, \{\leq\})$ be inequational specifications, $i = 0, 1$. Let $\mathcal{S} = (\Sigma, \mathcal{A}, \{\leq\}) = \mathcal{S}_0 \oplus \mathcal{S}_1$ be defined. Let $T_i = (\Sigma_i, D_i)$ be term deduction systems and let $T = (\Sigma, D) = T_0 \oplus T_1$ be defined. Let ξ be a preorder definable exclusively in terms of predicate and relation symbols. Let $\mathcal{A}_0$ be a complete axiomatisation with respect to the model induced by ξ in T_0 and let $\mathcal{A}$ be a sound axiomatisation with respect to the model induced by ξ in T. If T is an operational conservative extension up to ξ of T_0, then $\mathcal{S}$ is an inequational conservative extension of $\mathcal{S}_0$.*

We can deduce a general completeness theorem from Theorem 4.14. Therefore, we need the notion of elimination which roughly states that operators in an extended algebraic system can be expressed in the original system.

Definition 4.16 Let $\mathcal{S} = (\Sigma, \mathcal{A}, \mathcal{P})$ be an algebraic conservative extension of $\mathcal{S}_0 = (\Sigma_0, \mathcal{A}_0, \mathcal{P}_0)$. $p \in \mathcal{P}$ is an *elimination predicate* if for all $s \in C(\Sigma) \backslash C(\Sigma_0)$ there is a $t \in C(\Sigma_0)$ such that $\mathcal{A} \vdash p(s,t)$ and $\mathcal{A} \vdash p(t,s)$, i.e., for every new term there is a "p-symmetric" old term.

$\mathcal{S}$ has the *elimination property* if all predicates in P_0 are elimination predicates. □

Notice that this definition of elimination subsumes the definition of elimination on equational specifications (see for instance [BV95]) and the definition of elimination on inequational specifications [D'A95].

Theorem 4.17 *Under the same hypotheses of Theorem 4.14, if in addition $\mathcal{P} = \mathcal{P}_0$, all predicates in $\mathcal{P}$ are transitive, and $\mathcal{S}$ has the elimination property, then $\mathcal{A}$ is a complete axiomatisation of* A *with respect to $\mathcal{R}$.*

Proof. Let $s, t \in C(\Sigma_0)$ such that $\mathsf{A}/_{\mathcal{R}} \models p(s,t)$. Since $\mathsf{A}/_{\mathcal{R}}$ is a model conservative extension of $\mathsf{A}_0/_{\mathcal{R}_0}$ according to g with $g(p_{A_0}) = p_A$, then $\mathsf{A}_0/_{\mathcal{R}_0} \models p(s,t)$. So $\mathcal{A}_0 \vdash p(s,t)$ because $\mathcal{A}_0$ is complete, which trivially implies $\mathcal{A} \vdash p(s,t)$.

Suppose $s, t \in C(\Sigma) \backslash C(\Sigma_0)$ such that $\mathsf{A}/_{\mathcal{R}} \models p(s,t)$. Because $\mathcal{S}$ has the elimination property, there are $s', t' \in C(\Sigma_0)$ such that $\mathcal{A} \vdash p(s,s')$, $\mathcal{A} \vdash p(s',s)$, $\mathcal{A} \vdash p(t,t')$, and $\mathcal{A} \vdash p(t',t)$. Since $\mathcal{A}$ is sound, $\mathsf{A}/_{\mathcal{R}} \models p(s,s')$, $\mathsf{A}/_{\mathcal{R}} \models p(s',s)$, $\mathsf{A}/_{\mathcal{R}} \models p(t,t')$, and $\mathsf{A}/_{\mathcal{R}} \models p(t',t)$. Because p is transitive and $\mathcal{A}$ is sound, p_A is transitive, then $\mathsf{A}/_{\mathcal{R}} \models p(s',t')$. Now, since $\mathsf{A}/_{\mathcal{R}}$ is a model conservative extension of $\mathsf{A}_0/_{\mathcal{R}_0}$ according to g with $g(p_{A_0}) = p_A$, then $\mathsf{A}_0/_{\mathcal{R}_0} \models p(s',t')$. Because $\mathcal{A}_0$ is complete, $\mathcal{A}_0 \vdash p(s',t')$. Since $\mathcal{S}$ is an algebraic conservative extension of $\mathcal{S}_0$, $\mathcal{A} \vdash p(s',t')$. Finally, because p is transitive $\mathcal{A} \vdash p(s,t)$.

The proof of the cases of s and t belonging separately to $C(\Sigma_0)$ and $C(\Sigma) \backslash C(\Sigma_0)$ follows the same lines of the previous case omitting the considerations of elimination when s or t belongs to $C(\Sigma_0)$. $\square$

Assume that $\mathcal{S}$ is an inequational specification, we have that $\mathcal{P} = \mathcal{P}_0 = \{\leq\}$ and moreover $\leq$ is transitive. Thus, as an immediate corollary of the previous theorem we have the following important subcase. See Section 6 for many applications.

Theorem 4.18 *Under the same hypotheses of Theorem 4.15, if in addition $\mathcal{S}$ has the elimination property, $\mathcal{A}$ is a complete axiomatisation with respect to the model induced by the preorder ξ in T.*

Recall that an equational specification is an inequational specification with an additional conditional axiom (see Definition 3.3). We obtain as a trivial corollary that if $\mathcal{S}$, $\mathcal{S}_0$ and $\mathcal{S}_1$ are equational specifications and ξ is an equivalence, under the same conditions of Theorem 4.15, $\mathcal{S}$ is an equational conservative extension of $\mathcal{S}_0$. If moreover $\mathcal{S}$ has the elimination property, $\mathcal{A}$ is a complete axiomatisation with respect to the model induced by ξ in T.

Example 4.19 We demonstrate our results with the running example (see Section 6 for more information).

Lemma 3.8 states that BPA$^{\leq}$ is complete with respect to $\subsetneqq$ and PA$^{\leq}$ is sound with respect to $\subsetneqq$. Since, in addition, PA is an operational conservative extension up to $\subsetneqq$ of BPA, PA$^{\leq}$ is an inequational conservative extension of BPA$^{\leq}$. Moreover, because PA$^{\leq}$ has the elimination property (see [BW90]), it is a complete axiomatisation with respect to $\subsetneqq$ of PA.

Analogously, since BPA is a complete axiomatisation with respect to $\underline{\leftrightarrow}$ and $\mathrm{BPA_{dt}}$ is sound with respect to $\underline{\leftrightarrow}$, and moreover, the term deduction system of $\mathrm{BPA_{dt}}$ is an operational conservative extension of the term deduction system of BPA up to $\underline{\leftrightarrow}$ we may conclude that $\mathrm{BPA_{dt}}$ is an equational conservative extension of BPA. $\square$

5 Ground and confluence modulo equations

In this section we will use our main results to prove a theorem in term rewriting analysis. Therefore, we restrict ourselves to equational specifications in this section. We will prove a general reduction theorem stating that in many cases checking the Church-Rosser property for closed terms modulo some equations for a large system reduces to verifying this property for a small basic system. Of course, provided that the large system is an equational conservative extension of the small system. From a term rewriting point of view this condition is not realistic since usually the Church-Rosser property for closed terms is necessary to obtain conservativity.

In the previous section, we showed that, under certain conditions, it is possible to prove conservativity without a term rewriting analysis. Thus, we could argue that conservativity and ground confluence are equally powerful properties, so to speak.

Definition 5.1 A *term rewriting system* is a pair (Σ, R) with Σ a signature and R a set of *rewrite rules*. Rewrite rules are pairs of terms (over Σ) that we denote $s \to t$. We suppose that s is not a variable and that $var(t) \subseteq var(s)$. The *one step rewrite relation* $\to^1_R$ is the smallest relation on terms containing R that is closed under substitutions and contexts. The *rewrite relation* $\to_R$ is the transitive-reflexive closure of the one step rewrite relation $\to^1_R$. Often, we refer to a term rewriting system (Σ, R) by its set of (rewrite) rules R. $\square$

Definition 5.2 Let R be a set of rewrite rules and E be a set of equations. Let $=_E$ be the smallest congruence generated by the equations in E. The one step rewriting relation $\to^1_{R/E}$ is defined as $=_E \circ \to^1_R \circ =_E$. The rewriting relation $\to_{R/E}$ is the transitive-reflexive closure of the one step rewrite relation $\to^1_{R/E}$. (Recall that for two relations R and S we have $R \circ S = \{(r, s) \mid \exists t : (r, t) \in R, (t, s) \in S\}$.) $\square$

Definition 5.3 Let R be a set of rules and let E be a set of equations. Let $\to$ be the rewriting relation $\to_{R/E}$. Let s be a term. If for all s_0, s_1 such that $s \to s_0$ and $s \to s_1$ there is a term s' such that $s_0 \to s'$ and $s_1 \to s'$ we say that the rewriting relation $\to_{R/E}$ is *Church-Rosser* or *confluent*. We call $\to_{R/E}$ *ground Church-Rosser* if it is Church-Rosser for closed terms. Sometimes, we will write CR instead of Church-Rosser. We also say that $\to_R$ is Church-Rosser (or confluent) modulo E; we write CR$/E$. In the literature we also see E-Church-Rosser and E-confluence if $\to_{R/E}$ is confluent in the above sense; see, for instance, Jouannaud and Muñoz [JM84]. $\square$

Definition 5.4 Let R be a set of rules and let E be a set of equations. Let $=$ be the least congruence generated by the equations in E and the rules in R in both ways. We say that the rewriting relation $\to_{R/E}$ is ground CR$^=$ if for all ground terms s and t such that $s = t$ there are terms s' and t' such that $s \twoheadrightarrow_{R/E} s'$, $t \twoheadrightarrow_{R/E} t'$, and $s' =_E t'$. $\square$

Remark 5.5 It is not hard to see that $\to_R$ is ground CR/E if and only if it is ground CR$^=/E$. $\square$

Definition 5.6 A term rewriting system R is *(strongly) terminating* if there exists no infinite sequence $s_0 \to^1_R s_1 \to^1_R s_2 \ldots$. We call a term s a *normal form* if we do not have $s \to^1_R s'$ for any s'. $\square$

Theorem 5.7 *Let $\mathcal{S}_i = (\Sigma_i, \mathcal{A}_i, \{=\})$ be equational specifications. Let $\mathcal{S} = (\Sigma, \mathcal{A}, \{=\})$ $= \mathcal{S}_0 \oplus \mathcal{S}_1$ be defined. Suppose that $\mathcal{S}$ is an equational conservative extension of $\mathcal{S}_0$. Turn a set $R_0^= \subseteq \mathcal{A}_0$ into a set of rewrite rules R_0 and let $E = \mathcal{A}_0 \setminus R_0^=$ be a set of equations (or axioms). Turn the set $R^= = (\mathcal{A} \setminus E) \cup R_0^=$ into a set of rewrite rules R. Suppose that $\to_R$ is terminating and that normal forms are Σ_0 terms (so $\mathcal{S}$ has the elimination property). If $\to_{R_0/E}$ is ground Church-Rosser then $\to_{R/E}$ is also ground Church-Rosser.*

Proof. Let s and t be ground Σ terms and suppose that $\mathcal{A} \vdash s = t$. By assumption, there are ground Σ_0 terms s' and t' with $s \to_R s'$ and $t \to_R t'$. So $\mathcal{A} \vdash s' = t'$. Since $\mathcal{S}$ is a conservative extension of $\mathcal{S}_0$ we now have that $\mathcal{A}_0 \vdash s' = t'$. Since $\to_{R_0/E}$ is ground CR there are s_0 and t_0 such that $s' \to_{R_0/E} s_0$, $t' \to_{R_0/E} t_0$, and $E \vdash s_0 = t_0$. Since $R_0 \subseteq R$ we also have $s' \to_{R/E} s_0$. Since $s \to_R s'$ we also have $s \to_{R/E} s'$ (simply put $s =_E s, \ldots, s' =_E s'$ between the one step rewritings). So we find that $s \to_{R/E} s_0$. In the same way we find that $t \to_{R/E} t_0$, and we have $E \vdash s_0 = t_0$. This implies using remark 5.5 that $\to_{R/E}$ is ground CR. $\square$

In section 6 we will apply the just derived results.

6 Applications

In this section we will give the reader an idea of the applicability of our conservativity results, the completeness corollary and the ground Church-Rosser reduction theorem. Noteworthy perhaps, is that we could not find any conservativity result in the literature for which our method does not work, as well.

Within the ACP community there is a long tradition with conservativity results, completeness results and confluence results. Also in ATP there are many conservativity and completeness results. We will simultaneously treat numerous examples from ACP, ATP, and CCS. We will treat some typical cases more elaborately. We note that the examples in Figures 1, 2, 3, and 4 contain both known results and new results.

6.1 Applications in equational specifications

In the introduction we mentioned the problems concerning the confluence of ACP that Bergstra and Klop [BK84] used to prove conservativity. We claimed that with our theorems it is very easy to see that the conservativity result holds. Therefore, we elaborately treat the •-labelled arrow from ACP to BPA_δ in Figure 1. We show that all our general results apply to this arrow.

Van Glabbeek [Gla87] gives an operational semantics for Bergstra and Klop's ACP [BK84] and for their sequential subsystem BPA_δ [BK84]. With our operational result 4.8 it is easily seen that the large semantics is an operational conservative extension of the small one. Baeten and Weijland [BW90], for instance, show that BPA_δ is sound and complete with respect to the small semantics and that ACP is sound with respect to the large one. They use a variant of strong bisimulation with successful termination predicates, which is definable in terms of transition relations and predicates only. So, our equational result 4.15 immediately implies that ACP is an equational conservative extension of BPA_δ. Since ACP has the elimination property we also find the completeness of ACP with Theorem 4.18. Moreover, with our reduction Theorem 5.7 we have that the question whether or not ACP is ground Church-Rosser modulo associativity and commutativity of the choice (CR/AC) reduces to this question for BPA_δ. The associated term rewriting system of BPA_δ consists of five rewrite rules and two equations, which is a considerable reduction since the term rewriting system for ACP has many more rules.

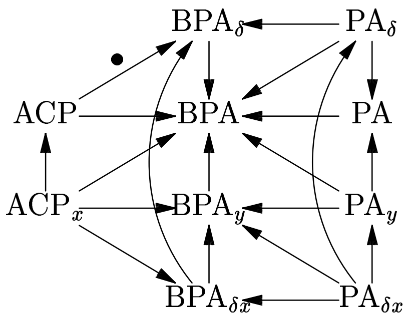

Figure 1: Applications in ACP

Now, we discuss Figure 1. An arrow $A \longrightarrow B$ indicates that system A is both an operational and an equational conservative extension of system B and that this can be shown using our conservativity results. The x and y stand for variables; we use them to treat many examples at the same time.

Let $x = y$. And let x be one of PR, RN, λ, Λ, θ, or a combination of them. The abbreviations stand for projections, renamings, simple state operators, extended state operators, and the priority operator respectively. A concise reference to these notions, their operational rules, their axiomatisations, and their associated term rewriting systems is the text book of Baeten and Weijland [BW90] or the survey [BV95]. The variant of bisimulation that is used in these applications is definable in terms of transition relations and predicates exclusively. So, for all these cases we have that all arrows of Figure 1: operational and equational conservativity. Moreover, all these extensions have the elimination property for either the complete BPA or the complete BPA_δ (if the extension contains already a δ); for full proofs see, for instance, [BW90] or [BV95]. So we find for all these extensions the completeness with Theorem 4.18. Moreover, the ground confluence modulo AC for these systems reduces to the ground confluence modulo AC for either BPA or BPA_δ.

Now, let $x = y$ and let x be one of rec, dt, or a combination of those (note that we can also combine rec with the already treated notions). The abbreviations stand for recursion and discrete time [BB92], respectively. Also for these systems we have that all arrows hold. Note that $\mathrm{BPA}_{\mathrm{dt}} \longrightarrow \mathrm{BPA}$ was one of the running examples. We do not have the elimination property for subscripted systems to systems without a subscript (for instance $\sigma_d(a)$ cannot be written as a BPA term). For the other arrows we have the elimination property [BB92], so from the completeness of BPA_y we conclude the completeness for all the extensions. The ground confluence of these systems has not yet been studied but with our reduction theorem it is only necessary to study the ground confluence for the BPA_y systems.

Now, let $x = y$ and let x be Milner's silent action τ. We already mentioned in the introduction that systems containing the three τ laws of Milner have in general bad rewriting properties. The conservativity of ACP_τ over ACP was proved semantically by Bergstra and Klop [BK85] since the second and third τ law have no clear term rewriting direction. Next, we will show that our approach also works in cases where the established method breaks down. In fact, we immediately find this result. The operational semantics of ACP_τ is just the one of ACP but now a ranges also over τ itself. It is easy to see that the conditions of Theorem 4.8 are satisfied, so ACP_τ is an operational conservative extension of ACP. Now with Theorem 4.7 we find that ACP_τ is an operational conservative extension up to rooted τ bisimulation equivalence of ACP. Since ACP is sound and complete and since ACP_τ is sound with respect to this equivalence, we find with Theorem 4.15 that ACP_τ is an equational conservative extension of ACP. All the other arrows in Figure 1 go likewise. Since all the extensions have the elimination property for $\mathrm{BPA}_{\delta,\tau}$, we find their completeness with the aid of the completeness of $\mathrm{BPA}_{\delta,\tau}$. The systems have bad term rewriting properties so the ground confluence results does not apply.

We mentioned in the introduction the rather complicated term rewriting analysis of Akkerman and Baeten [AB90] of a fragment of ACP with the branching τ. We will show in a moment that our results can be easily applied to this case. With the aid of Theorem 4.7 we find that ACP_τ is an operational conservative extension up to branching bisimulation equivalence [GW89] of ACP. Also in this case we find in the same way as above that ACP with the branching τ axioms [GW89], denoted ACP^τ, is an equational conservative extension of ACP. The same holds for all the other arrows in Figure 1. Since all the extensions have the elimination property for BPA^τ we find the completeness for them with the completeness of BPA^τ. The branching τ axioms have better term rewriting properties [AB90] than the τ laws of Milner (that we discussed above). So our ground confluence result may be useful, as well.

Let $x = y$ be the empty process ε of Koymans and Vrancken [KV85]; see also Vrancken [Vra91]. We can show operational and equational conservativity for all arrows from a system with an ε to a subsystem also featuring this ε by using the operational semantics that can be found in Baeten and Weijland's text book [BW90]. In [BW90] we also find that these systems have the elimination property, so also our completeness and the ground confluence results apply. For the remaining arrows we have to follow a different approach. The operational semantics in [BW90] features the rule $a \xrightarrow{a} \varepsilon$ so we can never have that this semantics is an operational conservative extension of a semantics without ε (but containing a). For, there is no ε in the subsystem. The solution to this problem is to take another operational semantics that is easily obtained by "upgrading" the complete graph model of Koymans and Vrancken [KV85]. In fact, this operational semantics is that of the subsystem where we include ε as a normal atomic action. So we have, for instance, $\varepsilon \xrightarrow{\varepsilon} \surd$. The special behaviour of the empty process is expressed with the aid of so-called ε bisimulation equivalence of Koymans and Vrancken [KV85]. Also this definition needs a straightforward upgrade from graphs to transitions (and is definable in terms of transition relations and predicates only). In this way we find the operational and equational conservativity. Since we cannot eliminate the empty process, we cannot apply our completeness theorem and the ground confluence result for these particular systems.

Let x be ρ standing for absolute real time [BB91]. Then the x-arrow in the figure holds. To obtain this result we take the operational semantics of Klusener [Klu91]. Also here we have the elimination property, so our completeness and ground confluence results apply, too.

$$\begin{tikzcd}
 & \mathrm{ATP} \arrow[dl] \arrow[d] & \mathrm{ATP}_u \arrow[l] \arrow[dl] \arrow[d] & \mathrm{ATP}_v \arrow[l] \arrow[dl] \arrow[d] \\
\mathrm{ASP} & \mathrm{ASTP} \arrow[l] & \mathrm{ASTP}_u \arrow[l] & \mathrm{ASTP}_v \arrow[l]
\end{tikzcd}$$

Figure 2: Applications in ATP

Now we treat results on ATP which are depicted in Figure 2. The acronym ASP stands for the algebra of sequential processes. This system stems from Milner [Mil84]. Nicollin and Sifakis [NS91] studied a timed process algebra called ATP with various extensions and restrictions of which the most restricted timed one is ASTP, the algebra of sequential timed processes. Milner's [Mil84] algebra of sequential processes ASP—the untimed version of ASTP—is the most restricted system. The interesting thing here is that they prove some conservativity results with the same strategy as ours: they show that the extensions are operationally conservative up to bisimulation by looking at the transition rules and then conclude the equational conservativity. Our figure intends to show that every possible extension that can be obtained with the so-called delay operators of Nicollin and Sifakis [NS91] is conservative. There are four delay operators present in [NS91]: start delay, unbounded start delay, execution delay, and termination delay. The termination delay (td) is an enhancement of the execution delay (ed) so if we have the termination delay we also have execution delay. For u we can take any combination of delay operators. If u does not contain all delay operators yet we can take for v the operators of u and a new one, or if the execution delay operator is in u we can take the termination delay operator in v and we do not necessarily need an extra delay operator for a non-trivial extension. Cases like ASP $\longrightarrow$ ASTP and $\mathrm{ASTP}_{\mathrm{td}} \longrightarrow \mathrm{ASTP}_{\mathrm{ed}}$ are, in our opinion, the most interesting since in these cases not only a new operator (unit delay and a special constant respectively) is introduced but also an original operator gets a new rule. Since the elimination property holds [NS91] for ASTP our completeness corollary applies for all the arrows but the two to ASP. The ground confluence of ATP is not yet studied but its study reduces to that of ASTP with our reduction theorem.

6.2 Applications in inequational specifications

$$\begin{tikzcd}
\mathrm{BPA}^*_\delta aa^{\leq} \arrow[d] & \mathrm{ACP}^* aa^{\leq} \arrow[l, dashed, "?"'] \arrow[dl] \arrow[d] \\
\mathrm{BPA}_\delta aa^{\leq} & \mathrm{ACP} aa^{\leq} \arrow[l]
\end{tikzcd}$$

Figure 3: Applications in $\mathrm{ACP}aa^{\leq}$

Voorhoeve and Basten introduced in [VB96] a preorder for unstable nondeterminism. They deal with a set of *autonomous actions* which can be regarded as observable actions that somehow behave as the silent step. Several algebras were defined there. $\mathrm{BPA}_\delta aa^{\leq}$ is the basic process algebra with deadlock and autonomous actions. They use our results to extend $\mathrm{BPA}_\delta aa^{\leq}$ with the parallel operator, obtaining thus $\mathrm{ACP}aa^{\leq}$. Moreover, since $\mathrm{ACP}aa^{\leq}$ has the elimination property, completeness is proved using our results. In addition, they added the binary Kleene star [BBP94] to both theories. Since $\mathrm{BPA}^*_\delta aa^{\leq}$ and $\mathrm{ACP}^* aa^{\leq}$ are sound, and the respective term deduction systems satisfy the conditions of Theorem 4.8, operational and inequational conservative extension can be also shown using our results. Figure 3 shows this overview. Perhaps, the reader expected the arrow $\mathrm{ACP}^* aa^{\leq} \longrightarrow \mathrm{BPA}^*_\delta aa^{\leq}$. In this case only operational conservative extension can be proved using results in this articles (and so operational conservative extension up to the preorder). Since $\mathrm{BPA}^*_\delta aa^{\leq}$ is not complete (see [Sew94, VB96]), Theorem 4.15 cannot be used.

The next application is based on Walker [Wal90]. He introduced in [Wal90] a complete (but non finite) axiomatisation for a preorder that extends τ bisimulation with divergence. Also here, we can use our results to prove conservativity and completeness.

$$
\begin{tikzcd}
\mathrm{ST}_\perp \arrow[d, "\text{new}"] & \mathrm{CCS}_\perp \arrow[l] \arrow[dl, "\text{new}"] \arrow[d, "\text{new}"] \\
\mathrm{ST} & \mathrm{CCS} \arrow[l]
\end{tikzcd}
$$

Figure 4: Applications in CCS

Below we will explain Figure 4. Let ST be the algebra of synchronisation trees with Milner's τ laws [Mil89]. The signature of ST has prefixing operators, the alternative composition and the *nil* process. Let CCS be the well-known calculus of Milner [Mil89] that extends ST with renaming, restriction and parallel composition, and the expansion laws. Let $\mathrm{ST}_\perp$ and $\mathrm{CCS}_\perp$ the respective extensions of ST and CCS with the divergence operator as given in [Wal90]. We note that for all CCS terms Walker's preorder agrees with rooted τ bisimulation [Wal90]. In addition, since ST is complete for the preorder, and the new operators can be eliminated, we can use our results to show that CCS is complete. Analogously, $\mathrm{CCS}_\perp$ is complete since $\mathrm{ST}_\perp$ is complete and $\mathrm{CCS}_\perp$ has the elimination property. Nevertheless, nor $\mathrm{ST}_\perp$ neither $\mathrm{CCS}_\perp$ have the elimination property with respect to ST or CCS. Moreover, it deserves to notice that the new-labelled arrows in Figure 4 are new results here.

7 Concluding remarks

In this paper we presented general conservativity results for transition system based process theories with reasonable and easy-to-check conditions. As a simple corollary of the conservativity results we proved a completeness theorem. We proved a general theorem giving sufficient conditions to reduce the question of ground confluence modulo some equations for a large term rewriting system associated with a process theory to a small term rewriting system under the condition that the large system is a conservative extension of the small one. With numerous examples that we took from the literature about CCS, ACP, and ATP we showed that our theorems are useful. The applications include various real and discrete time settings in ACP, ATP, and CCS and the notions projection, renaming, state operator, priority, recursion, the silent step (both the weak and branching variants), autonomous actions, the empty process, divergence, etc.

Remarkably, we could not find any conservativity results in the literature for which our method cannot be applied. We want to stress that the established method for proving conservativity in these theories usually makes use of a rather complicated term rewriting analysis, whereas our method is very easily applicable. This is a great advantage of our approach in our opinion.

Acknowledgements. This paper came into being by merging two earlier papers. One by Verhoef [Ver94] and one by D'Argenio [D'A95]. Wan Fokkink suggested us to merge those papers. We thank him for this idea and for his comments on the penultimate version of the ensuing amalgamation. We also thank Alban Ponse for being so helpful on the last version of this article. Moreover, we like to thank Gertjan Akkerman, Jos Baeten, Twan Basten, Jean-Pierre Jouannaud, Pim Kars, Jan Willem Klop, Xavier Nicollin, and the referees for their valuable comments on earlier versions of this paper and/or the other two.

References

[AB90] G.J. Akkerman and J.C.M. Baeten. Term rewriting analysis in process algebra. Report P9006, Programming Research Group, University of Amsterdam, 1990.

[Abr87] S. Abramsky. Observation equivalence as a testing equivalence. *Theoretical Computer Science*, 53:225–241, 1987.

[ABV92] L. Aceto, B. Bloom, and F.W. Vaandrager. Turning SOS rules into equations. In *Proceedings 7^{th} Annual Symposium on Logic in Computer Science,* Santa Cruz, California, pages 113–124. IEEE Computer Society Press, 1992.

[Ace91] L. Aceto. On relating concurrency and nondeterminism. In S. Brookes, M. Main, A. Melton, M. Mislove, and D. Schmidt, editors, *Proceedings Mathematical Foundations of Programming Semantics (MFPS 7)*, volume 598 of *Lecture Notes in Computer Science*, pages 376–402, Pittsburgh, 1991. Springer-Verlag.

[BB91] J.C.M. Baeten and J.A. Bergstra. Real time process algebra. *Journal of Formal Aspects of Computing Science*, 3(2):142–188, 1991.

[BB92] J.C.M. Baeten and J.A. Bergstra. Discrete time process algebra. In W.R. Cleaveland, editor, *Proceedings CONCUR 92,* Stony Brook, NY, USA, volume 630 of *Lecture Notes in Computer Science*, pages 401–420. Springer-Verlag, 1992.

[BB93] J.C.M. Baeten and J.A. Bergstra. Non interleaving process algebra. In Best [Bes93], pages 308–323.

[BBP94] J.A. Bergstra, I. Bethke, and A. Ponse. Process algebra with iteration and nesting. *The Computer Journal*, 37(4):243–258, 1994.

[BC88] G. Boudol and I. Castellani. Concurrency and atomicity. *Theoretical Computer Science*, 59(1/2):25–84, 1988.

[Bes93] E. Best, editor. *Proceedings CONCUR 93,* Hildesheim, Germany, volume 715 of *Lecture Notes in Computer Science*. Springer-Verlag, 1993.

[BG91] R.N. Bol and J.F. Groote. The meaning of negative premises in transition system specifications (extended abstract). In J. Leach Albert, B. Monien, and M. Rodríguez, editors, *Proceedings* 18^{th} *ICALP,* Madrid, volume 510 of *Lecture Notes in Computer Science*, pages 481–494. Springer-Verlag, 1991. Full version appeared as Report CS-R9054, CWI, Amsterdam, 1990. To appear in *Journal of the ACM.*

[BIM88] B. Bloom, S. Istrail, and A.R. Meyer. Bisimulation can't be traced: Preliminary report. In *Conference Record of the* 15^{th} *ACM Symposium on Principles of Programming Languages,* San Diego, California, pages 229–239, 1988.

[BIM95] B. Bloom, S. Istrail, and A.R. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42(1):232–268, January 1995.

[BK84] J.A. Bergstra and J.W. Klop. Process algebra for synchronous communication. *Information and Computation*, 60(1/3):109–137, 1984.

[BK85] J.A. Bergstra and J.W. Klop. Algebra of communicating processes with abstraction. *Theoretical Computer Science*, 37(1):77–121, 1985.

[BK90] J.C.M. Baeten and J.W. Klop, editors. *Proceedings CONCUR 90,* Amsterdam, volume 458 of *Lecture Notes in Computer Science*. Springer-Verlag, 1990.

[Bos94] D.J.B. Bosscher. Term rewriting properties of SOS axiomatisations. Report CS-R9405, CWI, Amsterdam, January 1994.

[BV95] J.C.M. Baeten and C. Verhoef. Concrete process algebra. In S. Abramsky, D. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science, Vol 4*. Oxford University Press, 1995.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra*. Cambridge Tracts in Theoretical Computer Science 18. Cambridge University Press, 1990.

[Cas93] I. Castellani. Observing distribution in processes. In *Proceedings MFCS'93*, volume 711 of *Lecture Notes in Computer Science*. Springer-Verlag, 1993.

[D'A95] P.R. D'Argenio. A general conservative extension theorem in process algebras with inequalities. In Ponse et al. [PVV95], pages 67–79.

[FV95] W.J. Fokkink and C. Verhoef. A conservative look at term deduction systems with variable binding. Report 95/28, Department of Mathematics and Computing Science, Eindhoven University of Technology, October 1995. Also as Logic Group Preprint Series 140, Department of Philosophy, Utrecht University, September 1995.

[FZ94] W.J. Fokkink and H. Zantema. Basic process algebra with iteration: Completeness of its equational axioms. *The Computer Journal*, 37(4):259–267, 1994.

[Gil76] A. Gill. *Applied Algebra for the Computer Sciences.* Prentice-Hall International, New Jersey – USA, 1976.

[Gla87] R.J. van Glabbeek. Bounded nondeterminism and the approximation induction principle in process algebra. In F.J. Brandenburg, G. Vidal-Naquet, and M. Wirsing, editors, *Proceedings STACS 87*, volume 247 of *Lecture Notes in Computer Science*, pages 336–347. Springer-Verlag, 1987.

[Gla90] R.J. van Glabbeek. The linear time – branching time spectrum. In Baeten and Klop [BK90], pages 278–297.

[Gla93] R.J. van Glabbeek. The linear time – branching time spectrum II (the semantics of sequential systems with silent moves). In Best [Bes93], pages 66–81.

[Gla95] R.J. van Glabbeek. The meaning of negative premises in transition system specifications II. Report STAN-CS-TN-95-17, Dept. of Computer Science, Stanford University, 1995. To appear in *Information and Computation.*

[Gro93] J.F. Groote. Transition system specifications with negative premises. *Theoretical Computer Science*, 118(2):263–299, 1993.

[GV92] J.F. Groote and F.W. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, October 1992.

[GW89] R.J. van Glabbeek and W.P. Weijland. Branching time and abstraction in bisimulation semantics (extended abstract). In G.X. Ritter, editor, *Information Processing 89*, pages 613–618. North-Holland, 1989. Full version available as Report CS-R9120, CWI, Amsterdam, 1991.

[Hen88] M. Hennessy. *Algebraic Theory of Processes.* MIT Press, Cambridge, Massachusetts, 1988.

[JM84] J.-P. Jouannaud and M. Muñoz. Termination of a set of rules modulo a set of equations. In R.E. Shostak, editor, *7th International Conference on Automated Deduction*, volume 170 of *Lecture Notes in Computer Science*, pages 175–193. Springer-Verlag, 1984.

[Klu91] A.S. Klusener. Completeness in real time process algebra. In J.C.M. Baeten and J.F. Groote, editors, *Proceedings CONCUR 91,* Amsterdam, volume 527 of *Lecture Notes in Computer Science*, pages 376–392. Springer-Verlag, 1991.

[Klu93] A.S. Klusener. *Models and axioms for a fragment of real time process algebra.* PhD thesis, Department of Mathematics and Computing Science, Eindhoven University of Technology, December 1993.

[KV85] C.P.J. Koymans and J.L.M. Vrancken. Extending process algebra with the empty process ϵ. Logic Group Preprint Series Nr. 1, CIF, State University of Utrecht, 1985.

[Mil84] R. Milner. A complete inference system for a class of regular behaviours. *Journal of Computer and System Sciences*, 28:439–466, 1984.

[Mil89] R. Milner. *Communication and Concurrency.* Prentice-Hall International, Englewood Cliffs, 1989.

[MT90] F. Moller and C. Tofts. A temporal calculus of communicating systems. In Baeten and Klop [BK90], pages 401–415.

[NS91] X. Nicollin and J. Sifakis. The algebra of timed processes ATP: Theory and application (revised version). Technical Report RT-C26, LGI-IMAG, Grenoble, France, November 1991.

[NT84] M. Nielsen and P.S. Thiagarajan. Degrees of non-determinism and concurrency: a Petri net view. In M. Joseph and R. Shyamasundar, editors, *Proceedings* 5^{th} *Conference on Foundations of Software Technology and Theoretical Computer Science,* India, volume 181 of *Lecture Notes in Computer Science*, pages 89–118. Springer-Verlag, 1984.

[Par81] D.M.R. Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, 5^{th} *GI Conference*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer-Verlag, 1981.

[Plo81] G.D. Plotkin. A structural approach to operational semantics. Report DAIMI FN-19, Computer Science Department, Aarhus University, 1981.

[PVV95] A. Ponse, C. Verhoef, and S.F.M. van Vlijmen, editors. *Proceedings of the 2nd. Workshop ACP'95*, Report 95/14. Department of Mathematics and Computing Science, Eindhoven University of Technology, 1995.

[Sew94] P. Sewell. Bisimulation is not (first order) equationally axiomatisable. In *Proceedings* 9^{th} *Annual Symposium on Logic in Computer Science,* Paris, France, pages 62–70. IEEE Computer Society Press, 1994.

[VB96] M. Voorhoeve and T. Basten. Process algebra with autonomous actions. Report 96/01, Department of Mathematics and Computing Science, Eindhoven University of Technology, 1996. A preliminary version appeared in Ponse et al. [PVV95].

[Ver94] C. Verhoef. A general conservative extension theorem in process algebra. In E.-R. Olderog, editor, *Proceedings IFIP Conference of Programming Concepts, Methods and Calculi,* San Miniato, Italy, volume A-56 of *IFIP Transactions*, pages 149–168. Elsevier Science Publishers, 1994.

[Ver95] C. Verhoef. A congruence theorem for structured operational semantics with predicates and negative premises. *Nordic Journal of Computing*, 2:274–302, 1995.

[Vra91] J.L.M. Vrancken. *Studies in process algebra, algebraic specifications and parallelism.* PhD thesis, University of Amsterdam, 1991.

[Wal90] D.J. Walker. Bisimulation and divergence. *Information and Computation*, 85(2):202–241, 1990.

[Wec92] W. Wechler. *Universal Algebra for Computer Scientists.* Springer-Verlag, Berlin, Germany, 1992.

[Yan93] D. Yankelevich. *Parametric views of process description languages.* PhD thesis, University of Pisa, 1993.

[Yi90] W. Yi. Real-time behaviour of asynchronous agents. In Baeten and Klop [BK90], pages 502–520.

[Zan95] H. Zantema. Termination of term rewriting by semantic labelling. *Fundamenta Informaticae*, 24:89–105, 1995.