On infinite guarded recursive specifications in process algebra

van Glabbeek, R.J.; Middelburg, C.A.

DOI
10.48550/arXiv.2005.00746
Publication date
2020
Document Version
Final published version
License
Unspecified

Link to publication

Citation for published version (APA):
van Glabbeek, R. J., & Middelburg, C. A. (2020). *On infinite guarded recursive specifications in process algebra*. (v1 ed.) ArXiv. https://doi.org/10.48550/arXiv.2005.00746

On Infinite Guarded Recursive Specifications in Process Algebra

R.J. van Glabbeek[1,2] and C.A. Middelburg[3]

[1] Data61, CSIRO, Sydney, Australia
[2] School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
rvg@cs.stanford.edu
[3] Informatics Institute, Faculty of Science, University of Amsterdam, Amsterdam, the Netherlands
C.A.Middelburg@uva.nl

Abstract. In most presentations of ACP with guarded recursion, recursive specifications are finite or infinite sets of recursion equations of which the right-hand sides are guarded terms. The completeness with respect to bisimulation equivalence of the axioms of ACP with guarded recursion has only been proved for the special case where recursive specifications are finite sets of recursion equations of which the right-hand sides are guarded terms of a restricted form known as linear terms. In this note, we widen this completeness result to the general case.

Keywords: process algebra, guarded recursion, completeness, infinitary conditional equational logic.

1998 ACM Computing Classification: F.1.2, F.4.1

1 Introduction

In ACP with guarded recursion, guarded recursive specifications, i.e. sets of recursion equations of which the right-hand sides are guarded terms, are used for recursive definitions of processes (see e.g. [1]). In most cases where ACP or a variant of it is extended with guarded recursion, guarded recursive specifications may be infinite. Moreover, countably infinite guarded recursive specifications are used in many applications of the process algebras concerned. Nevertheless, the completeness with respect to bisimulation equivalence of the axioms of ACP with guarded recursion has only been proved for the special case where recursive specifications are finite sets of recursion equations of which the right-hand sides are guarded terms of a restricted form known as linear terms.

The second author of this note realized in March 2017 that the completeness proof given in [2] for the above-mentioned special case could be widened to the general case. He communicated this at the time with several colleagues and forgot about it until it was recently mentioned in [3]. This mention motivated him to write a note about the general completeness result. It is noteworthy that the proof of the fact on which the widening of the existing completeness proof is based (Theorem 1) turned out to be less straightforward than the second author thought in March 2017 and comes from the first author.

In order to make this note self-contained, it contains short surveys of ACP and its extension with guarded recursion. We did not attach much importance to preventing any text overlap with surveys from earlier papers.

2 Algebra of Communicating Processes

In this section, we give a survey of ACP (Algebra of Communicating Processes). For a comprehensive overview of ACP, the reader is referred to [1,2].

In ACP, it is assumed that a fixed but arbitrary set A of *actions*, with $\delta \notin \mathsf{A}$, has been given. We write A_δ for $\mathsf{A} \cup \{\delta\}$. It is further assumed that a fixed but arbitrary commutative and associative *communication* function $\gamma : \mathsf{A}_\delta \times \mathsf{A}_\delta \to \mathsf{A}_\delta$, with $\gamma(\delta, a) = \delta$ for all $a \in \mathsf{A}_\delta$, has been given. The function γ is regarded to give the result of synchronously performing any two actions for which this is possible, and to give δ otherwise.

The signature of ACP consists of the following constants and operators:

- for each $a \in \mathsf{A}$, the *action* constant a ;
- the *inaction* constant δ ;
- the binary *alternative composition* operator $\_ + \_$;
- the binary *sequential composition* operator $\_ \cdot \_$;
- the binary *parallel composition* operator $\_ \parallel \_$;
- the binary *left merge* operator $\_ \mathbin{|\!\!\lfloor} \_$;
- the binary *communication merge* operator $\_ \mid \_$;
- for each $H \subseteq \mathsf{A}$, the unary *encapsulation* operator ∂_H .

We assume that there is an infinite set $\mathcal{X}$ of variables which contains x, y and z with and without subscripts. Terms over the signature of ACP, also referred to as ACP terms, are built as usual. We use infix notation for the binary operators. The precedence conventions used with respect to the operators of ACP are as follows: $+$ binds weaker than all others, $\cdot$ binds stronger than all others, and the remaining operators bind equally strong.

The constants of ACP can be explained as follows ($a \in \mathsf{A}$):

- δ denotes the process that cannot do anything;
- a denotes the process that first performs action a and after that terminates successfully.

Let t and t' be closed ACP terms denoting processes p and p'. Then the operators of ACP can be explained as follows:

- $t + t'$ denotes the process that behaves as p or behaves as p' (but not both);
- $t \cdot t'$ denotes the process that first behaves as p and on successful termination of p next behaves as p';
- $t \parallel t'$ denotes the process that behaves as p and p' in parallel;

Table 1. Axioms of ACP

$x + y = y + x$		A1	$x \parallel y = x \mathbin{⌊\!⌊} y + y \mathbin{⌊\!⌊} x + x \mid y$	CM1
$(x + y) + z = x + (y + z)$		A2	$a \mathbin{⌊\!⌊} x = a \cdot x$	CM2
$x + x = x$		A3	$a \cdot x \mathbin{⌊\!⌊} y = a \cdot (x \parallel y)$	CM3
$(x + y) \cdot z = x \cdot z + y \cdot z$		A4	$(x + y) \mathbin{⌊\!⌊} z = x \mathbin{⌊\!⌊} z + y \mathbin{⌊\!⌊} z$	CM4
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$		A5	$a \cdot x \mid b = (a \mid b) \cdot x$	CM5
$x + \delta = x$		A6	$a \mid b \cdot x = (a \mid b) \cdot x$	CM6
$\delta \cdot x = \delta$		A7	$a \cdot x \mid b \cdot y = (a \mid b) \cdot (x \parallel y)$	CM7
			$(x + y) \mid z = x \mid z + y \mid z$	CM8
$\partial_H(a) = a$	if $a \notin H$	D1	$x \mid (y + z) = x \mid y + x \mid z$	CM9
$\partial_H(a) = \delta$	if $a \in H$	D2		
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$		D3		
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$		D4	$a \mid b = \gamma(a, b)$	CF

- $t \mathbin{⌊\!⌊} t'$ denotes the same process as $t \parallel t'$, except that it starts with performing an action of p;
- $t \mid t'$ denotes the same process as $t \parallel t'$, except that it starts with performing an action of p and an action of p' synchronously;
- $\partial_H(t)$ denotes the process that behaves the same as p, except that actions from H are blocked.

The operators $\mathbin{⌊\!⌊}$ and $\mid$ are of an auxiliary nature. They are needed to axiomatize ACP.

The axioms of ACP are the equations given in Table 1. In these equations, a and b stand for arbitrary constants of ACP, and H stands for an arbitrary subset of A. In D1 and D2, side conditions restrict what a and H stand for.

In the sequel, we will use the sum notation $\sum_{i<n} t_i$. Let $t_0, t_1, t_2, \ldots$ be terms over the signature of ACP or an extension of ACP. Then $\sum_{i<0} t_i = \delta$ and, for each $n \in \mathbb{N}$ with $n > 0$, the term $\sum_{i<n} t_i$ is defined by induction on n as follows: $\sum_{i<1} t_i = t_0$ and $\sum_{i<n+1} t_i = \sum_{i<n} t_i + t_n$.

3 ACP with Guarded Recursion

In this section, we give a survey of the extension of ACP with guarded recursion. For a comprehensive overview of guarded recursion in the setting of ACP, the reader is referred to [1,2].

A closed ACP term denotes a process with a finite upper bound to the number of actions that it can perform. Guarded recursion allows the description of processes without a finite upper bound to the number of actions that it can perform.

Table 2. Axioms for guarded recursion

$\langle X\|E\rangle = \langle t\|E\rangle$	if $X = t \in E$	RDP
$E \Rightarrow X = \langle X\|E\rangle$	if $X \in \mathrm{V}(E)$	RSP

Let t be a term over the signature of ACP or an extension of ACP in which a variable X occurs. Then an occurrence of X in t is *guarded* if t has a subterm of the form $a \cdot t'$ where $a \in \mathsf{A}$ and t' contains this occurrence of X. An ACP term t is a guarded ACP term if all occurrences of variables in t are guarded.

A *guarded recursive specification* over ACP is a set $\{X_i = t_i \mid i \in I\}$, where I is a finite or infinite set, each X_i is a variable from $\mathcal{X}$, each t_i is either a guarded ACP term in which only variables from $\{X_i \mid i \in I\}$ occur or an ACP term rewritable to such a term using the axioms of ACP in either direction and/or the equations in $\{X_j = t_j \mid j \in I \land i \neq j\}$ from left to right, and $X_i \neq X_j$ for all $i, j \in I$ with $i \neq j$.

We write $\mathrm{V}(E)$, where E is a guarded recursive specification, for the set of all variables that occur in E. The equations occurring in a guarded recursive specification are called *recursion equations*.

A solution of a guarded recursive specification E in some model of ACP is a set $\{p_X \mid X \in \mathrm{V}(E)\}$ of elements of the carrier of that model such that each equation in E holds if, for all $X \in \mathrm{V}(E)$, X is assigned p_X. We are only interested in models of ACP in which guarded recursive specifications have unique solutions.

We extend ACP with guarded recursion by adding constants for solutions of guarded recursive specifications over ACP and axioms concerning these additional constants. For each guarded recursive specification E over ACP and each $X \in \mathrm{V}(E)$, we add a constant $\langle X|E\rangle$ that stands for the unique solution of E for X to the constants of ACP. We add the equation RDP and the conditional equation RSP given in Table 2 to the axioms of ACP. In RDP and RSP, X stands for an arbitrary variable from $\mathcal{X}$, t stands for an arbitrary ACP term, E stands for an arbitrary guarded recursive specification over ACP, and the notation $\langle t|E\rangle$ is used for t with, for all $X \in \mathrm{V}(E)$, all occurrences of X in t replaced by $\langle X|E\rangle$. Side conditions restrict what X, t and E stand for. We write $\mathrm{ACP}_{\mathrm{rec}}$ for the resulting theory. Terms over the signature of $\mathrm{ACP}_{\mathrm{rec}}$ are also referred to as $\mathrm{ACP}_{\mathrm{rec}}$ terms.

The equations $\langle X|E\rangle = \langle t|E\rangle$ and the conditional equations $E \Rightarrow X = \langle X|E\rangle$ for a fixed E express that the constants $\langle X|E\rangle$ make up a solution of E and that this solution is the only one.

Because we have to deal with conditional equational formulas with an infinite number of premises in $\mathrm{ACP}_{\mathrm{rec}}$, it is understood that infinitary conditional equational logic is used in deriving equations from the axioms of $\mathrm{ACP}_{\mathrm{rec}}$. A complete inference system for infinitary conditional equational logic can be found in [4]. It is noteworthy that in the case of infinitary conditional equational logic derivation trees may be infinitely branching (but they may not have infinite branches).

We write $T \vdash t = t'$, where T is ACP or $\mathrm{ACP}_{\mathrm{rec}}$, to indicate that the equation $t = t'$ is derivable from the axioms of T using a complete inference system for infinitary conditional equational logic.

4 Linear Recursive Specifications

In this section, we show that each guarded recursive specification over ACP can be reduced to one in which the right-hand sides of recursion equations are guarded terms of a restricted form known as linear terms. This result will be used in Section 6. In its proof, we make use of the fact that each guarded ACP term has a head normal form.

Let T be ACP or $\mathrm{ACP}_{\mathrm{rec}}$. The set *HNF* of *head normal forms of* T is inductively defined by the following rules:

– $\delta \in \mathit{HNF}$;
– if $a \in \mathsf{A}$, then $a \in \mathit{HNF}$;
– if $a \in \mathsf{A}$ and t is a term over the signature of T, then $a \cdot t \in \mathit{HNF}$;
– if $t, t' \in \mathit{HNF}$, then $t + t' \in \mathit{HNF}$.

Each head normal form of T is derivably equal to a head normal form of the form $\sum_{i<n} a_i \cdot t_i + \sum_{j<m} b_i$, where $n, m \in \mathbb{N}$, for each $i < n$, $a_i \in \mathsf{A}$ and t_i is a term over the signature of T, and, for each $j < m$, $b_j \in \mathsf{A}$.

Each guarded $\mathrm{ACP}_{\mathrm{rec}}$ term is derivably equal to a head normal form of $\mathrm{ACP}_{\mathrm{rec}}$.

Proposition 1 (Head normal form). *For each guarded* $\mathrm{ACP}_{\mathrm{rec}}$ *term* t*, there exists a head normal form* t' *of* $\mathrm{ACP}_{\mathrm{rec}}$ *such that* $\mathrm{ACP}_{\mathrm{rec}} \vdash t = t'$.

Proof. First we prove the following weaker result about head normal forms:

> *For each guarded* ACP *term* t*, there exists a head normal form* t' *of* ACP *such that* $\mathrm{ACP} \vdash t = t'$.

The proof is straightforward by induction on the structure of t. The case where t is of the form δ and the case where t is of the form a ($a \in \mathsf{A}$) are trivial. The case where t is of the form $t_1 + t_2$ follows immediately from the induction hypothesis. The case where t is of the form $t_1 \cdot t_2$ follows immediately from the induction hypothesis and the claim that, for all head normal forms t_1 and t_2 of ACP, there exists a head normal form t' of ACP such that $t_1 \cdot t_2 = t'$ is derivable from the axioms of ACP. This claim is easily proved by induction on the structure of t_1. The cases where t is of one of the forms $t_1 \mathbin{⌊\!⌊} t_2$, $t_1 \mid t_2$ or $\partial_H(t_1)$ are proved along the same lines as the case where t is of the form $t_1 \cdot t_2$. In the case that t is of the form $t_1 \mid t_2$, each of the cases to be considered in the inductive proof of the claim demands a proof by induction on the structure of t_2. The case that t is of the form $t_1 \parallel t_2$ follows immediately from the case that t is of the form $t_1 \mathbin{⌊\!⌊} t_2$ and the case that t is of the form $t_1 \mid t_2$. Because t is a guarded ACP term, the case where t is a variable cannot occur.

The proof of the proposition itself is also straightforward by induction on the structure of t. The cases other than the case where t is of the form $\langle X|E\rangle$ is proved in the same way as in the above proof of the weaker result. The case where t is of the form $\langle X|E\rangle$ follows immediately from the weaker result and RDP. $\square$

The set LT of *linear* ACP *terms* is inductively defined by the following rules:

- $\delta \in LT$;
- if $a \in \mathsf{A}$, then $a \in LT$;
- if $a \in \mathsf{A}$ and $X \in \mathcal{X}$, then $a \cdot X \in LT$;
- if $t, t' \in LT$, then $t + t' \in LT$.

Clearly, each linear ACP term is also a guarded ACP term (but not vice versa).

A *linear recursive specification* over ACP is a guarded recursive specification E over ACP such that, for each equation $X = t \in E$, $t \in LT$.

Each guarded recursive specification over ACP can be reduced to a linear recursive specification over ACP.

Theorem 1 (Reduction). *For each guarded recursive specification E over* ACP *and each $X \in \mathrm{V}(E)$, there exists a finite or countably infinite linear recursive specification E' over* ACP *such that* $\mathrm{ACP}_{\mathrm{rec}} \vdash \langle X|E\rangle = \langle X|E'\rangle$.

Proof. We approach this algorithmically. In the construction of the linear recursive specification E', we keep a set V of recursion equations from E' that are already found and a sequence W of equations of the form $X_k = \langle t_k|E\rangle$ that still have to be transformed. The algorithm has a finite or countably infinite number of stages. In each stage, V and W are finite. Initially, V is empty and W contains only the equation $X_0 = \langle X|E\rangle$.

In each stage, we remove the first equation from W. Assume that this equation is $X_k = \langle t_k|E\rangle$. We bring the term $\langle t_k|E\rangle$ into head normal form. If t_k is not a guarded term, then we use RDP here to turn t_k into a guarded term first. Thus, by Proposition 1, we can always bring the term $\langle t_k|E\rangle$ into head normal form. Assume that the resulting head normal form is $\sum_{i<n} a_i \cdot t'_i + \sum_{j<m} b_j$. Then, we add the equation $X_k = \sum_{i<n} a_i \cdot X_{k+i+1} + \sum_{j<m} b_j$, where the X_{k+i+1} are fresh variables, to the set V. Moreover, for each $i < n$, we add the equation $X_{k+i+1} = t'_i$ to the end of the sequence W. Notice that the terms t'_i are of the form $\langle t_{k+i+1}|E\rangle$.

Because V grows monotonically, there exists a limit. That limit is the finite or countably infinite linear recursive specification E'. Every equation that is added to the finite sequence W, is also removed from it. Therefore, the right-hand side of each equation from E' only contains variables that also occur as the left-hand side of an equation from E'.

Now, we want to use RSP to show that $\mathrm{ACP}_{\mathrm{rec}} \vdash \langle X|E\rangle = \langle X|E'\rangle$. The variables occurring in E' are $X_0, X_1, X_2, \ldots$. For each k, the variable X_k has been exactly once in W as the left-hand side of an equation. For each k, assume that this equation is $X_k = \langle t_k|E\rangle$. To use RSP, we have to show for each k that the equation $X_k = \sum_{i<n} a_i \cdot X_{k+i+1} + \sum_{j<m} b_j$ from E' with, for each l, all

occurrences of X_l replaced by $\langle t_l|E\rangle$ is derivable from the axioms of $\mathrm{ACP_{rec}}$. For each k, this follows from the construction. $\square$

An immediate corollary of Theorem 1 is the following expressiveness result: in each model of $\mathrm{ACP_{rec}}$, the processes that can be described by a guarded recursive specification over ACP and the processes that can be described by a finite or countably infinite linear recursive specification over ACP are the same.

5 Semantics of ACP with Guarded Recursion

In this section, we present a structural operational semantics of $\mathrm{ACP_{rec}}$ and define a notion of bisimulation equivalence based on this semantics.

We start with presenting a structural operational semantics of $\mathrm{ACP_{rec}}$. The following relations on closed $\mathrm{ACP_{rec}}$ terms are used:

- for each $a \in \mathsf{A}$, a unary relation $\xrightarrow{a}\surd$;
- for each $a \in \mathsf{A}$, a binary relation $\xrightarrow{a}$.

We write $t \xrightarrow{a}\surd$ instead of $\xrightarrow{a}\surd(t)$ and $t \xrightarrow{a} t'$ instead of $\xrightarrow{a}(t,t')$. The relations $\xrightarrow{a}\surd$ and $\xrightarrow{a}$ can be explained as follows:

- $t \xrightarrow{a}\surd$: t can perform action a and then terminate successfully;
- $t \xrightarrow{a} t'$: t can perform action a and then behave as t'.

The structural operational semantics of $\mathrm{ACP_{rec}}$ is described by the rules given in Table 3. In these tables, a, b, and c stand for arbitrary actions from A, X stands for an arbitrary variable from $\mathcal{X}$, t stands for an arbitrary ACP term, and E stands for an arbitrary guarded recursive specification over ACP.

A *bisimulation* is a binary relation R on closed $\mathrm{ACP_{rec}}$ terms such that, for all closed $\mathrm{ACP_{rec}}$ terms t_1, t_2 with $R(t_1, t_2)$, the following conditions hold:

- if $t_1 \xrightarrow{a} t'_1$, then there exists a closed $\mathrm{ACP_{rec}}$ term t'_2 such that $t_2 \xrightarrow{a} t'_2$ and $R(t'_1, t'_2)$;
- if $t_2 \xrightarrow{a} t'_2$, then there exists a closed $\mathrm{ACP_{rec}}$ term t'_1 such that $t_1 \xrightarrow{a} t'_1$ and $R(t'_1, t'_2)$;
- $t_1 \xrightarrow{a}\surd$ iff $t_2 \xrightarrow{a}\surd$.

Two closed $\mathrm{ACP_{rec}}$ terms t_1, t_2 are *bisimulation equivalent*, written $t_1 \leftrightarroweq t_2$, if there exists a bisimulation R such that $R(t_1, t_2)$.

Proposition 2 (Congruence). *$\leftrightarroweq$ is a congruence with respect to the operators of* $\mathrm{ACP_{rec}}$.

The axioms of $\mathrm{ACP_{rec}}$ are sound with respect to bisimulation equivalence for equations between closed terms.

Theorem 2 (Soundness). *For all closed* $\mathrm{ACP_{rec}}$ *terms t and t', $t \leftrightarroweq t'$ if* $\mathrm{ACP_{rec}} \vdash t = t'$.

The proofs of Proposition 2 and Theorem 2 can, for example, be found in [1].

Table 3. Rules for the operational semantics of $\mathrm{ACP}_{\mathrm{rec}}$

$$\frac{}{a \xrightarrow{a} \surd}$$

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

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

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

$$\frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} \surd}{x \parallel y \xrightarrow{c} \surd}\ \gamma(a,b) = c \qquad \frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} y'}{x \parallel y \xrightarrow{c} y'}\ \gamma(a,b) = c$$

$$\frac{x \xrightarrow{a} x',\ y \xrightarrow{b} \surd}{x \parallel y \xrightarrow{c} x'}\ \gamma(a,b) = c \qquad \frac{x \xrightarrow{a} x',\ y \xrightarrow{b} y'}{x \parallel y \xrightarrow{c} x' \parallel y'}\ \gamma(a,b) = c$$

$$\frac{x \xrightarrow{a} \surd}{x \mathbin{⌊\!⌊} y \xrightarrow{a} y} \qquad \frac{x \xrightarrow{a} x'}{x \mathbin{⌊\!⌊} y \xrightarrow{a} x' \parallel y}$$

$$\frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} \surd}{x \mid y \xrightarrow{c} \surd}\ \gamma(a,b) = c \qquad \frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} y'}{x \mid y \xrightarrow{c} y'}\ \gamma(a,b) = c$$

$$\frac{x \xrightarrow{a} x',\ y \xrightarrow{b} \surd}{x \mid y \xrightarrow{c} x'}\ \gamma(a,b) = c \qquad \frac{x \xrightarrow{a} x',\ y \xrightarrow{b} y'}{x \mid y \xrightarrow{c} x' \parallel y'}\ \gamma(a,b) = c$$

$$\frac{x \xrightarrow{a} \surd}{\partial_H(x) \xrightarrow{a} \surd}\ a \notin H \qquad \frac{x \xrightarrow{a} x'}{\partial_H(x) \xrightarrow{a} \partial_H(x')}\ a \notin H$$

$$\frac{\langle t|E\rangle \xrightarrow{a} \surd}{\langle X|E\rangle \xrightarrow{a} \surd}\ X = t\ \in\ E \qquad \frac{\langle t|E\rangle \xrightarrow{a} x'}{\langle X|E\rangle \xrightarrow{a} x'}\ X = t\ \in\ E$$

6 Completeness of ACP with Guarded Recursion

It follows from Theorem 1 and the completeness proof given in [2] for the special case of finite linear recursive specifications over ACP that the axioms of $\mathrm{ACP}_{\mathrm{rec}}$ are also complete with respect to bisimulation equivalence for equations between closed terms.

Theorem 3 (Completeness). *For all closed* $\mathrm{ACP}_{\mathrm{rec}}$ *terms t and t', $t \leftrightarroweq t'$ only if* $\mathrm{ACP}_{\mathrm{rec}} \vdash t = t'$.

Proof. Theorem 4.4.1 from [2] states that, for all closed $\mathrm{ACP}_{\mathrm{rec}}$ terms t and t' in which only constants $\langle X|E\rangle$ occur where E is a finite linear recursive specification, $t \leftrightarroweq t'$ only if $\mathrm{ACP}_{\mathrm{rec}} \vdash t = t'$. We can strengthen this theorem by dropping the finiteness condition because the proof given in [2] does not rely on it. It follows immediately from the strengthened version of Theorem 4.4.1 from [2] and Theorem 1 from the current paper that, for all closed $\mathrm{ACP}_{\mathrm{rec}}$ terms t and t', $t \leftrightarroweq t'$ only if $\mathrm{ACP}_{\mathrm{rec}} \vdash t = t'$. $\square$

To the best of our knowledge, the completeness of the axioms of $\mathrm{ACP}_{\mathrm{rec}}$ with respect to bisimulation equivalence has as yet only been proved for the special case of finite linear recursive specifications. Crucial for the completeness for the general case is that infinitary conditional equational logic is used in deriving equations from the axioms of $\mathrm{ACP}_{\mathrm{rec}}$. The use of this logic is inescapable with infinite guarded recursive specifications. This speaks for itself, but it is virtually unmentioned in the literature on process algebra.

7 Concluding Remarks

We have widened the existing completeness result for $\mathrm{ACP}_{\mathrm{rec}}$. A by-product of this work is the following expressiveness result: in each model of $\mathrm{ACP}_{\mathrm{rec}}$, the processes that can be described by a guarded recursive specification over ACP and the processes that can be described by a finite or countably infinite linear recursive specification over ACP are the same. Notice that even uncountably infinite guarded recursive specifications over ACP can be reduced to finite or countably infinite linear recursive specifications over ACP.

References

1. Baeten, J.C.M., Weijland, W.P.: Process Algebra, Cambridge Tracts in Theoretical Computer Science, vol. 18. Cambridge University Press, Cambridge (1990)
2. Fokkink, W.J.: Introduction to Process Algebra. Texts in Theoretical Computer Science, An EATCS Series, Springer-Verlag, Berlin (2000)
3. van Glabbeek, R.J.: Failure trace semantics for a process algebra with time-outs. `arXiv:2002.10814v1 [cs.LO]` (2020)
4. van Glabbeek, R.J., Vaandrager, F.W.: Modular specification of process algebras. Theoretical Computer Science 113(2), 293–348 (1993)