Process Algebra with Five-Valued Conditions

Jan A. Bergstra[1,2] and Alban Ponse[1]

[1] University of Amsterdam, Programming Research Group, Kruislaan 403, NL-1098 SJ Amsterdam, The Netherlands.
`http://www.wins.uva.nl/research/prog/` `alban@wins.uva.nl`
[2] Utrecht University, Department of Philosophy, P.O. Box 80126, NL-3508 TC Utrecht, The Netherlands.
`http://www.phil.uu.nl/eng/home.html`

Abstract. We propose a five-valued logic that can be motivated from an algorithmic point of view and from a logical perspective. This logic is combined with process algebra. For process algebra with five-valued logic we present an operational semantics in SOS-style and a completeness result. Finally, we discuss some generalizations.

Key words & Phrases: Concurrency, process algebra, many-valued logic, conditional guard construct, conditional composition.

1991 CR Categories: F.3, F.4.3, I.1.

1 Introduction

Assume P is some simple program or algorithm. Then the initial behaviour of

$$\texttt{if}\ \phi\ \texttt{then}\ P\ \texttt{else}\ P$$

depends on evaluation of the condition ϕ: either it yields an immediate error, or it starts performing P, or it diverges in evaluation of ϕ. Note that the second possibility only requires that ϕ is either true or false. The following three non-classical truth values accommodate these intuitions:

Meaningless. Typical examples are errors that are detectable during execution such as a type-clash or division by zero.

Choice or *undetermined.* A typical example is *alternative composition,* i.e. in `if` ϕ `then` Q `else` P either P or Q is executed.

Divergent or *undefined.* Typically, evaluation of a partial predicate can diverge.

We describe a propositional logic that incorporates these three non-classical truth values and discuss its combination with process algebra. Here process algebra is used as a vehicle to specify and analyze concurrent algorithms: a (closed) process term is considered an algebraic notation for an algorithm. We shall use an `if_then_else_` construct in which the condition ranges over five-valued propositions. We end the paper with some generalizations and conclusions.

Acknowledgement. We thank Bas Luttik and Piet Rodenburg for discussion, proof reading, and verifying a completeness result (on $\mathbb{K}_4$).

2 Five-Valued Logic

First we shortly consider the incorporation of each of the previously mentioned non-classical truth values in classical two-valued logic. In [8] it is established that there are only two three-valued logics that satisfy the (nice) algebraic properties defined by the axioms in Table 1, where T stands for "true", F for "false", and $*$ denotes a "third truth value":

Kleene's three-valued logic $\mathbb{K}_3$. This three-valued logic, which we call $\mathbb{K}_3$, is introduced in [18] to model propositional combination of partial predicates. $\mathbb{K}_3$ is defined by the following truth tables:

x	$\neg x$
T	F
F	T
$*$	$*$

$\wedge$	T	F	$*$
T	T	F	$*$
F	F	F	F
$*$	$*$	F	$*$

$\vee$	T	F	$*$
T	T	T	T
F	T	F	$*$
$*$	T	$*$	$*$

and is characterized (cf. [8]) by the axioms in Table 1 and the absorption axiom

$$\text{(Abs)} \quad x \vee (x \wedge y) = x.$$

Strict three-valued logic $\mathbb{S}_3$. This three-valued is due to Bochvar [14]. Citing [8]: "Here, on the theory that one bad apple spoils the barrel, an expression has value $*$ as soon as it has a component with that value". $\mathbb{S}_3$ is defined by

x	$\neg x$
T	F
F	T
$*$	$*$

$\wedge$	T	F	$*$
T	T	F	$*$
F	F	F	$*$
$*$	$*$	$*$	$*$

$\vee$	T	F	$*$
T	T	T	$*$
F	T	F	$*$
$*$	$*$	$*$	$*$

According to [8], $\mathbb{S}_3$ is characterized by the axioms in Table 1 and axioms

$$\begin{array}{ll} \text{(S1)} & x \vee (\neg x \wedge y) = x \vee y, \\ \text{(S2)} & * \wedge x = *. \end{array}$$

The combination of these logics is studied in [8], which also comprises an account of McCarthy's asymmetric connectives.

Table 1. Axioms for three-valued logic.

(1)	$\neg \mathsf{T} = \mathsf{F}$	(5)	$x \wedge y = y \wedge x$
(2)	$\neg * = *$	(6)	$x \wedge (y \wedge z) = (x \wedge y) \wedge z$
(3)	$\neg\neg x = x$	(7)	$\mathsf{T} \wedge x = x$
(4)	$\neg(x \wedge y) = \neg x \vee \neg y$	(8)	$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$

We observe that two different intuitions for Kleene's non-classical truth value can be distinguished: *choice* or *undetermined*, further written as C, and *divergent* or *undefined*, denoted by D. Incorporation of both C and D leads to a four-valued logic that we call

$$\mathbb{K}_4$$

and that—as far as we know—has not been studied before. It can be argued that the axioms given for $\mathbb{K}_3$ allow at most two distinct elements that satisfy $\neg * = *$, and with C and D in this role imply the identity

$$\mathsf{C} \wedge \mathsf{D} = \mathsf{F}.$$

Adding this identity and replacing (2) in Table 1 by $\neg\mathsf{C} = \mathsf{C}$ and $\neg\mathsf{D} = \mathsf{D}$ yields with axiom (Abs) a complete axiomatization for $\mathbb{K}_4$ [20]. Note that $\mathbb{S}_3$ cannot be generalized in a similar fashion because of axiom (S2). Following [8] we set M, called *meaningless*, for the non-classical truth value occurring in $\mathbb{S}_3$.

Combining $\mathbb{S}_3$ and $\mathbb{K}_4$ yields a five-valued logic with constants in $\mathbb{T}_5 = \{\mathsf{M}, \mathsf{C}, \mathsf{T}, \mathsf{F}, \mathsf{D}\}$. In order to combine this logic with process algebra we shall add McCarthy's asymmetric connectives and conditional composition, and we shall incorporate *fluents* to represent "deterministic conditions".

Asymmetric connectives. With $\mathbin{_{\circ}\!\wedge}$ we denote McCarthy's left to right conjunction (cf. [21]), adopting the asymmetric notation from [8]. First the left argument is evaluated, and if necessary the right argument. From [8] and the intuitions provided for M, C, and D it follows that

$$c \mathbin{_{\circ}\!\wedge} x = c \text{ for } c \in \{\mathsf{M}, \mathsf{F}, \mathsf{D}\} \text{ and } c \mathbin{_{\circ}\!\wedge} x = c \wedge x \text{ for } c \in \{\mathsf{C}, \mathsf{T}\}.$$

With $\mathbin{^{\circ}\!\vee}$ we denote the dual of $\mathbin{_{\circ}\!\wedge}$, called left-sequential disjunction and defined by $x \mathbin{^{\circ}\!\vee} y = \neg(\neg x \mathbin{_{\circ}\!\wedge} \neg y)$. So in accordance with the intuition of sequential evaluation, logics with divergence D or meaningless M are asymmetric with respect to these connectives.

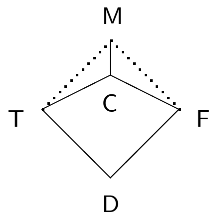
We now list the complete truth tables for $\neg$, $\wedge$, and $\mathbin{_{\circ}\!\wedge}$:

x	$\neg x$
M	M
C	C
T	F
F	T
D	D

$\wedge$	M	C	T	F	D
M	M	M	M	M	M
C	M	C	C	F	F
T	M	C	T	F	D
F	M	F	F	F	F
D	M	F	D	F	D

$\mathbin{_{\circ}\!\wedge}$	M	C	T	F	D
M	M	M	M	M	M
C	M	C	C	F	F
T	M	C	T	F	D
F	F	F	F	F	F
D	D	D	D	D	D

and we define disjunction $\vee$ as usual: $x \vee y = \neg(\neg x \wedge \neg y)$. We denote the resulting five-valued logic by $\Sigma_5(\neg, \wedge, \mathbin{_{\circ}\!\wedge})$, or shortly Σ_5. Note that the axioms from Table 1 are valid for Σ_5 (with $*$ ranging over $\{\mathsf{M}, \mathsf{C}, \mathsf{D}\}$) and that $\mathbin{_{\circ}\!\wedge}$ and its dual $\mathbin{^{\circ}\!\vee}$ are idempotent and associative. The five truth values in $\mathbb{T}_5$ can be arranged in the following partial ordering, reflecting information order and (argumentwise) monotony of $\wedge$ and $\mathbin{_{\circ}\!\wedge}$:



(The outer rhombus represents the original lattice from [8, 13], without C.)

Conditional composition. The expression $x \triangleleft y \triangleright z$, of which the notation stems from [17], denotes `if` y `then` x `else` z. Sequential connectives provide a useful intuition if conditional composition is introduced in the logic:

$$y \mathbin{{}_{\circ}\!\wedge} x = x \triangleleft y \triangleright \mathsf{F}.$$

This is plausible because it provides the very underlying intuition of $\mathbin{{}_{\circ}\!\wedge}$ (first evaluate y, then, if necessary, x). Similarly, we have $y \mathbin{{}^{\circ}\!\vee} x = \mathsf{T} \triangleleft y \triangleright x$. We first define $\_ \triangleleft \_ \triangleright \_$ as a ternary operation:

$$x \triangleleft \mathsf{M} \triangleright y = \mathsf{M}$$
$$x \triangleleft \mathsf{T} \triangleright y = x$$
$$x \triangleleft \mathsf{F} \triangleright y = y$$
$$x \triangleleft \mathsf{D} \triangleright y = \mathsf{D}$$

$\triangleleft$ C $\triangleright$	M	C	T	F	D
M	M	M	M	M	M
C	M	C	C	C	C
T	M	C	T	C	T
F	M	C	C	F	F
D	M	C	T	F	D

Notice that $x \triangleleft \mathsf{C} \triangleright y$ (as a binary operation) is idempotent, commutative, and associative. This operation can be defined by:

$$x \triangleleft \mathsf{C} \triangleright y = (\mathsf{C} \wedge x) \vee (\mathsf{C} \wedge y) \vee (x \wedge y).$$

Proposition 2.1. *Conditional composition $x \triangleleft y \triangleright z$ can be defined in Σ_5 by*

$$x \triangleleft y \triangleright z = ((y \vee \mathsf{D}) \mathbin{{}_{\circ}\!\wedge} (x \mathbin{{}^{\circ}\!\vee} \mathcal{G})) \triangleleft \mathsf{C} \triangleright ((\neg y \vee \mathsf{D}) \mathbin{{}_{\circ}\!\wedge} (z \mathbin{{}_{\circ}\!\wedge} \mathcal{H})),$$

where $x \triangleleft \mathsf{C} \triangleright y$ is given above, $\mathcal{G} = (y \mathbin{{}_{\circ}\!\wedge} x) \vee (\neg y \mathbin{{}_{\circ}\!\wedge} z)$, and $\mathcal{H} = (\neg y \mathbin{{}^{\circ}\!\vee} x) \wedge (y \mathbin{{}^{\circ}\!\vee} z)$.

Fluents. Following McCarthy and Hayes [23], let $f, g, \ldots$ be names for *fluents*, i.e., objects that in any state (i.e., at each instance of time) may take a deterministic value, thus a value in $\{\mathsf{M}, \mathsf{T}, \mathsf{F}, \mathsf{D}\}$. We write

$$f : \mathsf{DetFluent}$$

to express this, and $f : \mathsf{BoolFluent}$ if fluent f ranges over $\{\mathsf{T}, \mathsf{F}\}$. Fluents are used to model *deterministic conditions*, for example conditions that can occur in an algorithm or a program. Deterministic conditions are further considered in the next section. Let $\mathbb{P}_4$ be a set of fluents of type $\mathsf{DetFluent}$. We write

$$\Sigma_5(\mathbb{P}_4)$$

for the extension of Σ_5 with the fluents in $\mathbb{P}_4$, and we let $\Sigma_5(\mathbb{P}_2)$ denote the extension of Σ_5 with fluents of type $\mathsf{BoolFluent}$ in set $\mathbb{P}_2$. In order to equate conditions defined in $\Sigma_5(\mathsf{DetFluent})$ we use substitution of fluents:

$$[\phi/f]g \triangleq g, \qquad [\phi/f]c \triangleq c \text{ for } c \in \{\mathsf{M},\mathsf{C},\mathsf{T},\mathsf{F},\mathsf{D}\},$$
$$[\phi/f]f \triangleq \phi, \qquad [\phi/f]\neg\psi \triangleq \neg[\phi/f]\psi,$$
$$[\phi/f](\psi_1 \diamondsuit \psi_2) \triangleq [\phi/f]\psi_1 \diamondsuit [\phi/f]\psi_2 \text{ for } \diamondsuit \in \{\wedge, \mathbin{{}_{\circ}\!\wedge}\},$$

and as a proof rule the *excluded fifth rule* (cf. [13]):

$$\frac{\sigma(\phi) = \sigma(\psi) \qquad \text{for all } \sigma \in \{[\mathsf{M}/f],[\mathsf{T}/f],[\mathsf{F}/f],[\mathsf{D}/f]\}}{\phi = \psi}.$$

Together with the identities generated by the truth tables this yields a complete evaluation system for equations over $\Sigma_5(\mathbb{P}_4)$. With the associated *excluded third rule* (on substitution of T and F for fluents of type $\mathsf{BoolFluent}$) we find an evaluation system for $\Sigma_5(\mathbb{P}_2)$. We write

$$\Sigma_5(\mathbb{P}_4) \models \phi = \psi$$

if $\phi = \psi$ follows from the system defined above and the truth tables for Σ_5. The identity stated in the following lemma is used later on, and can be easily proved.

Lemma 2.2. $\Sigma_5(\mathbb{P}_4) \models \phi \vee \mathsf{D} = \phi \mathbin{{}^{\circ}\!\vee} \mathsf{D}$.

3 ACP with Five-Valued Conditions

The axiom system $\mathsf{ACP}(A,\gamma)$ (see e.g., [9, 10, 6]) is parameterized with a set A of constants $a, b, c, \ldots$ denoting atomic actions (atoms), i.e., processes that are not subject to further division, and that execute in finite time. In $\mathsf{ACP}(A,\gamma)$ there is a constant $\delta \notin A$, denoting the inactive process. We write A_δ for $A \cup \{\delta\}$. The six operations of $\mathsf{ACP}(A,\gamma)$ are

Sequential composition: $X \cdot Y$ denotes the process that performs X, and upon completion of X starts with Y.
Alternative composition: $X + Y$ denotes the process that performs either X or Y.
Merge or *parallel composition*: $X \parallel Y$ denotes the parallel execution of X and Y (including the possibility of synchronization).
Left merge, an auxiliary operator: $X \mathbin{\|\!\!\_} Y$ denotes $X \parallel Y$ with the restriction that the first action stems for the left argument X.
Communication merge, an auxiliary operator: $X \mid Y$ denotes $X \parallel Y$ with the restriction that the first action is a synchronization of both X and Y.
Encapsulation: $\partial_H(X)$ (where $H \subseteq A$) renames atoms in H to δ.

We mostly suppress the $\cdot$ in process expressions, and brackets according to the following rules: $\cdot$ binds strongest, and $\|$, $\llbracket$, $|$ all bind stronger than $+$.

In $\mathsf{ACP}(A,\gamma)$ the *communication function* $\gamma : A \times A \to A_\delta$ defines whether actions communicate, and if so, i.e., $\gamma(a,b) \neq \delta$, to what result. In Table 2 we present a slight modification of $\mathsf{ACP}(A,\gamma)$. This modification concerns commutativity of the communication merge $|$ (axiom (CMC), explaining the missing (CM6) and (CM9)). We set $|\restriction_{(A\times A)} = \gamma$.

Table 2. The axiom system $\mathsf{ACP}(A,\gamma)$, where $a,b,c \in A_\delta$, $H \subseteq A$.

(A1)	$X+(Y+Z)=(X+Y)+Z$	(CM1)	$X \parallel Y = (X \llfloor Y + Y \llfloor X) + X \mid Y$
(A2)	$X+Y=Y+X$	(CM2)	$a \llfloor X = aX$
(A3)	$X+X=X$	(CM3)	$aX \llfloor Y = a(X \parallel Y)$
(A4)	$(X+Y)Z = XZ+YZ$	(CM4)	$(X+Y) \llfloor Z = X \llfloor Z + Y \llfloor Z$
		(CMC)	$X \mid Y = Y \mid X$
(A5)	$(XY)Z = X(YZ)$	(CM5)	$aX \mid b = (a \mid b)X$
(A6)	$X+\delta = X$	(CM7)	$aX \mid bY = (a \mid b)(X \parallel Y)$
(A7)	$\delta X = \delta$	(CM8)	$(X+Y) \mid Z = X \mid Z + Y \mid Z$
		(D1)	$\partial_H(a) = a$ if $a \notin H$
(C1)	$a \mid b = b \mid a$	(D2)	$\partial_H(a) = \delta$ if $a \in H$
(C2)	$(a \mid b) \mid c = a \mid (b \mid c)$	(D3)	$\partial_H(X+Y) = \partial_H(X) + \partial_H(Y)$
(C3)	$\delta \mid a = \delta$	(D4)	$\partial_H(XY) = \partial_H(X)\partial_H(Y)$

A (very) simple $\mathsf{ACP}(A,\gamma)$ process term is $a \parallel b$, the *interleaving* of two atomic actions a,b, i.e., the setting in which $a \mid b = \delta$. It easily follows that

$$\mathsf{ACP}(A,\gamma) \vdash a \parallel b = ab + ba.$$

A key feature of process algebra is *conditional composition*

$$X \lhd \phi \rhd Y,$$

which represents `if` ϕ `then` X `else` Y where X,Y range over processes and ϕ is a condition. Its introduction in process algebra is described in [3]. In [11–13] we have extended the scope of the condition in conditional composition to various many-valued logics as described in [8], with the intention to model and analyze the occurrence of error-prone conditions in algorithms. Repeated use of conditional composition can lead to cumbersome notation, e.g.,

$$a_1 \cdot X_1 \lhd \phi_1 \rhd (a_2 \cdot X_2 \lhd \phi_2 \rhd (a_3 \cdot X_3 \lhd \phi_3 \rhd a_4 \cdot X_4)),$$

and to laborious inspection of the outer arguments of conditional composition (either processes or again conditions). Therefore we introduce the following alternative notation

$$X +_\phi Y = X \lhd \phi \rhd Y,$$

which has been borrowed from the conventions in probabilistic process algebra [5]. We use association to the right. The above term then reads as

$$a_1 \cdot X_1 +_{\phi_1} a_2 \cdot X_2 +_{\phi_2} a_3 \cdot X_3 +_{\phi_3} a_4 \cdot X_4,$$

which is easier to grasp. A condition in $\Sigma_5(\mathbb{P}_4)$ is called *deterministic* if it does not contain C. There is a fundamental difference between C and the other non-classical constants: the truth values M and D can be established by some external device (e.g., a type checker or a mathematician), whereas C is—on purpose—beyond any means of analysis. We only know it either behaves as T or as F. Of course, a process such as $ab + ba$ can also be described by $ab +_{\mathsf{C}} ba$ and, more generally, we may consider $+$ as a derived construct if C and conditional composition are available. Stated differently: the alternative composition $+$ of process algebra can be viewed as a notational device which allows one to remove the non-classical truth value C from process expressions involving atoms, sequential composition, and conditional composition (cf. Lemma 4.3).

Instead of conditional composition we shall often use the *conditional guard construct*

$$\phi :\rightarrow X,$$

which (roughly) expresses `if` ϕ `then` X. In Table 3 axioms are given for combining $\mathsf{ACP}(A, \gamma)$ with five-valued conditions. Here the constant μ represents the operational contents of M and was introduced in [11, 13]. Furthermore, the ϕ in the conditional guard construct ranges over $\Sigma_5(\mathbb{P}_4)$, so $\phi :\rightarrow$ is considered as a *unary* operation and related to conditional composition by axiom (Cond). Later on we show that $\phi :\rightarrow X = X \triangleleft \phi \triangleright \delta$. The conditional guard construct binds weaker than $\cdot$ and stronger than $\|$, $\llfloor$, and $|$.

Observe that the axioms (GC7) and (GC8) generalize (CM5) and (CM7), respectively. Also observe that $\phi :\rightarrow X \mid \psi :\rightarrow Y \neq \phi \wedge \psi :\rightarrow (X \mid Y)$ (set $\phi :\rightarrow X \equiv \mathsf{T} :\rightarrow \mu$ and $\psi :\rightarrow Y \equiv \mathsf{F} :\rightarrow \delta$). We use the acronym

$$\mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4)$$

both to refer to the axioms of Tables 2 and 3, and to the signature thus defined.

In order to combine process algebra and five-valued logic, we finally introduce the 'rule of equivalence'

$$\text{(ROE)} \quad \frac{\models \phi = \psi}{\vdash \phi :\rightarrow X = \psi :\rightarrow X}$$

This rule reflects the 'rule of consequence' in Hoare's Logic (cf. [1]). We write

$$\mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4) + \text{ROE}_5 \vdash X = Y,$$

or shortly $\vdash X = Y$, if $X = Y$ follows from the axioms of $\mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4)$, the axioms and rules for $\Sigma_5(\mathbb{P}_4)$, and the appropriate rule of equivalence

$$(\text{ROE}_5) \quad \frac{\Sigma_5(\mathbb{P}_4) \models \phi = \psi}{\mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4) \vdash \phi :\rightarrow X = \psi :\rightarrow X}$$

Table 3. Remaining axioms of $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)$, $a,b \in A_\delta$, $H \subseteq A$, and $\phi \in \Sigma_5(\mathbb{P}_4)$.

(M1)	$X + \mu = \mu$	(Cond)	$X \lhd \phi \rhd Y = \phi :\rightarrow X + \neg\phi :\rightarrow Y$
(M2)	$\mu \cdot X = \mu$	(GC1)	$\phi :\rightarrow X + \psi :\rightarrow X = \phi \vee \psi :\rightarrow X$
(M3)	$\mu \mid X = \mu$	(GC2)	$\phi :\rightarrow X + \phi :\rightarrow Y = \phi :\rightarrow (X + Y)$
		(GC3)	$(\phi :\rightarrow X)Y = \phi :\rightarrow XY$
		(GCL4)	$\phi :\rightarrow (\psi :\rightarrow X) = \phi \mathbin{\underset{\circ}{\wedge}} \psi :\rightarrow X$
(GM)	$\mathsf{M} :\rightarrow X = \mu$	(GC5)	$\phi :\rightarrow X \mathbin{\|\!\lfloor} Y = \phi :\rightarrow (X \mathbin{\|\!\lfloor} Y)$
(GC)	$\mathsf{C} :\rightarrow X = X$	(GC6)	$\phi :\rightarrow a \mid \psi :\rightarrow b = \phi \wedge \psi :\rightarrow a \mid b$
(GT)	$\mathsf{T} :\rightarrow X = X$	(GC7)	$\phi :\rightarrow aX \mid \psi :\rightarrow b = \phi \wedge \psi :\rightarrow (a \mid b)X$
(GF)	$\mathsf{F} :\rightarrow X = \delta$	(GC8)	$\phi :\rightarrow aX \mid \psi :\rightarrow bY = \phi \wedge \psi :\rightarrow (a \mid b)(X \parallel Y)$
(GD)	$\mathsf{D} :\rightarrow X = \delta$	(GC9)	$\partial_H(\phi :\rightarrow X) = \phi :\rightarrow \partial_H(X)$

We end this section with some useful derivabilities, applied in the remainder of the paper.

Lemma 3.1. 1. $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4) + \mathrm{ROE}_5 \vdash \phi :\rightarrow X = \phi \mathbin{\overset{\circ}{\vee}} \mathsf{D} :\rightarrow X$,
2. $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4) + \mathrm{ROE}_5 \vdash \phi \mathbin{\overset{\circ}{\vee}} \psi :\rightarrow X = \phi \vee (\neg\phi \mathbin{\underset{\circ}{\wedge}} \psi) :\rightarrow X$,
3. $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4) + \mathrm{ROE}_5 \vdash \phi \wedge \psi :\rightarrow X = (\phi \mathbin{\underset{\circ}{\wedge}} \psi) \vee (\psi \mathbin{\underset{\circ}{\wedge}} \phi) :\rightarrow X$.

Proof. As for 1. We apply ROE_5 on the identity proved in Lemma 2.2:

$$\phi :\rightarrow X = \phi :\rightarrow X + \delta = \phi :\rightarrow X + \mathsf{D} :\rightarrow X = \phi \vee \mathsf{D} :\rightarrow X \overset{2.2}{=} \phi \mathbin{\overset{\circ}{\vee}} \mathsf{D} :\rightarrow X.$$

As for 2 and 3. By inspection, taking all possible value-pairs for ϕ, ψ, and axioms (GM)–(GD). ∎

Using 3.1.1,2 and (Cond) one easily derives $\phi :\rightarrow X = X +_\phi \delta$.

4 Operational Semantics and Completeness

In this section we provide $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)$ with an operational semantics and come up with a completeness result. Of course, interpretations of the conditions occurring at 'top level' in a process expression also determine its semantics. As an example, consider for fluent f and action a the expression $f :\rightarrow a$. Depending on the interpretation of f, this process either behaves as μ, as a, or as δ.

Given a (non-empty) set $\mathbb{P}_4$ of fluents, let w range over $\mathcal{W}$, the *valuations* (interpretations) of $\mathbb{P}_4$ in $\{\mathsf{M},\mathsf{T},\mathsf{F},\mathsf{D}\}$. In the usual way we extend w to $\Sigma_5(\mathbb{P}_4)$:

$$\begin{aligned} w(c) &\overset{\triangle}{=} c \text{ for } c \in \{\mathsf{M},\mathsf{C},\mathsf{T},\mathsf{F},\mathsf{D}\}, \\ w(\neg\phi) &\overset{\triangle}{=} \neg(w(\phi)), \\ w(\phi \diamondsuit \psi) &\overset{\triangle}{=} w(\phi) \diamondsuit w(\psi) \text{ for } \diamondsuit \in \{\wedge, \mathbin{\underset{\circ}{\wedge}}\}. \end{aligned}$$

From the evaluation system defined in Section 2, it follows that

$$\forall w \in \mathcal{W}\big(\models w(\phi) = w(\psi)\big) \implies \models \phi = \psi.$$

In Table 4 we define for each $w \in \mathcal{W}$ a unary predicate *meaningless*, notation $\mu(w, \_)$, over process terms in $\mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4)$. This predicate defines whether a process expression represents the meaningless process μ under valuation w.

Table 4. Rules for $\mu(w, \_)$ in *panth*-format.

μ	$\mu(w,\mu)$	
$:\rightarrow$	$\mu(w, \phi :\rightarrow X)$ if $w(\phi) = \mathsf{M}$	$\dfrac{\mu(w,X)}{\mu(w, \phi :\rightarrow X)}$ if $w(\phi) \in \{\mathsf{C}, \mathsf{T}\}$
$+, \cdot, \|, \mathbin{\|\!\lfloor}, \mid, \partial_H$	$\dfrac{\mu(w,X)}{\begin{array}{c}\mu(w, X+Y)\\ \mu(w, Y+X)\\ \mu(w, X \cdot Y)\\ \mu(w, \partial_H(X))\end{array}}$	$\dfrac{\mu(w,X)}{\begin{array}{c}\mu(w, X \parallel Y)\\ \mu(w, Y \parallel X)\\ \mu(w, X \mathbin{\|\!\lfloor} Y)\\ \mu(w, X \mid Y)\end{array}}$

The axioms and rules for $\mu(w, \_)$ given in Table 4 are extended by axioms and rules given in Table 5, which define transitions

$$\_ \xrightarrow{w,a} \_ \subseteq \mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4) \times \mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4)$$

and unary "tick-predicates" or "termination transitions"

$$\_ \xrightarrow{w,a} \surd \subseteq \mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4)$$

for all $w \in \mathcal{W}$ and $a \in A$. Transitions characterize under which interpretations a process expression defines the possibility to execute an atomic action, and what remains to be executed (if anything, otherwise $\surd$ symbolizes successful termination).

The axioms and rules in Tables 4 and 5 yield a structured operational semantics (SOS) with negative premises in the style of Groote [16]. Moreover, they satisfy the so called *panth-format* defined by Verhoef [24] and define the following notion of bisimulation equivalence:

Definition 4.1. *Let $B \subseteq \mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4) \times \mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4)$. Then B is a bisimulation if for all P, Q with PBQ the following conditions hold for all $w \in \mathcal{W}$ and $a \in A$:*

- $\mu(w, P) \iff \mu(w, Q)$,
- $\forall P' \; (P \xrightarrow{w,a} P' \implies \exists Q' (Q \xrightarrow{w,a} Q' \wedge P'BQ'))$,
- $\forall Q' \; (Q \xrightarrow{w,a} Q' \implies \exists P' (P \xrightarrow{w,a} P' \wedge P'BQ'))$,
- $P \xrightarrow{w,a} \surd \iff Q \xrightarrow{w,a} \surd$.

Two processes P, Q are bisimilar, notation $P \leftrightarrows Q$, if there exists a bisimulation containing the pair (P, Q).

Table 5. Transition rules in *panth*-format.

$a \in A$	$a \xrightarrow{w,a} \surd$	
$\cdot, \lfloor\!\lfloor$	$\dfrac{X \xrightarrow{w,a} \surd}{X \cdot Y \xrightarrow{w,a} Y \quad X \lfloor\!\lfloor Y \xrightarrow{w,a} Y}$	$\dfrac{X \xrightarrow{w,a} X'}{X \cdot Y \xrightarrow{w,a} X'Y \quad X \lfloor\!\lfloor Y \xrightarrow{w,a} X' \parallel Y}$
$+, \parallel$	$\dfrac{X \xrightarrow{w,a} \surd \quad \neg\mu(w,Y)}{X+Y \xrightarrow{w,a} \surd \quad Y+X \xrightarrow{w,a} \surd \quad X \parallel Y \xrightarrow{w,a} Y \quad Y \parallel X \xrightarrow{w,a} Y}$	$\dfrac{X \xrightarrow{w,a} X' \quad \neg\mu(w,Y)}{X+Y \xrightarrow{w,a} X' \quad Y+X \xrightarrow{w,a} X' \quad X \parallel Y \xrightarrow{w,a} X' \parallel Y \quad Y \parallel X \xrightarrow{w,a} Y \parallel X'}$
$a \mid b = c$	$\dfrac{X \xrightarrow{w,a} \surd \quad Y \xrightarrow{w,b} \surd}{X \mid Y \xrightarrow{w,c} \surd \quad X \parallel Y \xrightarrow{w,c} \surd}$ $a \mid b = c$	$\dfrac{X \xrightarrow{w,a} \surd \quad Y \xrightarrow{w,b} Y'}{X \mid Y \xrightarrow{w,c} Y' \quad X \parallel Y \xrightarrow{w,c} Y'}$ $a \mid b = c$
	$\dfrac{X \xrightarrow{w,a} X' \quad Y \xrightarrow{w,b} \surd}{X \mid Y \xrightarrow{w,c} X' \quad X \parallel Y \xrightarrow{w,c} X'}$ $a \mid b = c$	$\dfrac{X \xrightarrow{w,a} X' \quad Y \xrightarrow{w,b} Y'}{X \mid Y \xrightarrow{w,c} X' \parallel Y' \quad X \parallel Y \xrightarrow{w,c} X' \parallel Y'}$ $a \mid b = c$
∂_H	$\dfrac{X \xrightarrow{w,a} \surd}{\partial_H(X) \xrightarrow{w,a} \surd}$ if $a \notin H$	$\dfrac{X \xrightarrow{w,a} X'}{\partial_H(X) \xrightarrow{w,a} \partial_H(X')}$ if $a \notin H$
$:\rightarrow$	$\dfrac{X \xrightarrow{w,a} \surd}{\phi :\rightarrow X \xrightarrow{w,a} \surd}$ if $w(\phi) \in \{\mathsf{C}, \mathsf{T}\}$	$\dfrac{X \xrightarrow{w,a} X'}{\phi :\rightarrow X \xrightarrow{w,a} X'}$ if $w(\phi) \in \{\mathsf{C}, \mathsf{T}\}$

Furthermore, from [16, 24] it easily follows that the transitions and meaningless instances defined by these axioms and rules are uniquely determined. This can be established with help of the following *stratification* S:

$$S(\mu(w,X)) = 0,\ \ S(X \xrightarrow{w,a} X') = S(X \xrightarrow{w,a} \surd) = 1.$$

By the main result in [24] it follows that bisimilarity is a *congruence* relation for all operations involved. Notice that conditional guard constructs are considered here as unary operations: for each $\phi \in \Sigma_5(\mathbb{P}_4)$ there is an operation $\phi :\rightarrow \_$.

We write $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)/_{\leftrightarrows} \models P = Q$ whenever $P \leftrightarrows Q$ according to the notions just defined, and for $\boldsymbol{X} = X_1,...,X_n$

$$\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)/_{\leftrightarrows} \models t_1(\boldsymbol{X}) = t_2(\boldsymbol{X})$$

if for all $\boldsymbol{P} = P_1,...,P_n$ it holds that $t_1(\boldsymbol{P}) = t_2(\boldsymbol{P})$. It is not difficult, but tedious to establish that in the bisimulation model thus obtained all equations of Table 2 are true. Hence we conclude:

Lemma 4.2. *The system* $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4) + \mathrm{ROE}_5$ *is sound with respect to bisimulation: if* $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4) + \mathrm{ROE}_5 \vdash t_1(\boldsymbol{X}) = t_2(\boldsymbol{X})$*, then*

$$\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)/_{\leftrightarrows} \models t_1(\boldsymbol{X}) = t_2(\boldsymbol{X}).$$

Finally, we provide a completeness result for $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)+\mathrm{ROE}_5$. Our proof refers to the completeness result in [13], which is based on a representation of closed process terms for which bisimilarity implies derivability in a straightforward way (so called “basic terms”). A crucial observation is that terms over $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)$ can be represented without C.

Lemma 4.3. *In* $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)$ *each closed process expression can be proved equal to one in which* C *does not occur.*

Proof. We omit a full proof based on a representation of closed terms not containing ∂_H, $\|$, $\lfloor\!\lfloor$, $|$, and $\_ \lhd \_ \rhd \_$ (both as a logical connective and as a process constructor, cf. Proposition 2.1). It can be argued that C need not occur in any guard ϕ in $\phi :\rightarrow X$ by induction on the complexity of ϕ. E.g., if $\phi \equiv \phi_1 \wedge \phi_2$ then by Lemma 3.1.3, $\phi :\rightarrow X = (\phi_1 \,{}_{\circ}\!\!\wedge\, \phi_2) :\rightarrow X + (\phi_2 \,{}_{\circ}\!\!\wedge\, \phi_1) :\rightarrow X = \phi_1 :\rightarrow (\phi_2 :\rightarrow X) + \phi_2 :\rightarrow (\phi_1 :\rightarrow X)$. ∎

Theorem 4.4. *The system* $\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4) + \mathrm{ROE}_5$ *is complete with respect to bisimulation: for closed terms* P *and* Q*,*

$$\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4) + \mathrm{ROE}_5 \vdash P = Q \iff \mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)/_{\leftrightarrows} \models P \leftrightarrows Q.$$

Proof. By the previous lemma and soundness it is sufficient to prove $\Longleftarrow$ for $\mathsf{ACP}(A,\gamma)$ with four-valued logic over $\{\mathsf{M},\mathsf{T},\mathsf{F},\mathsf{D}\}$ and $\mathbb{P}_4$. A detailed (inductive) proof is spelled out in [13]. ∎

We end this section with a nice correspondence result.

Proposition 4.5. *Let $t_1(\boldsymbol{X}, \boldsymbol{x}) = t_2(\boldsymbol{X}, \boldsymbol{x})$ be a process identity with process variables $\boldsymbol{X}$ and condition variables $\boldsymbol{x}$ in which the only constants are in Σ_5 and the only operation is $\_ \triangleleft \_ \triangleright \_$. Then*

$$\mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4)/_{\leftrightarrows} \models t_1(\boldsymbol{X}, \boldsymbol{x}) = t_2(\boldsymbol{X}, \boldsymbol{x}) \Longleftrightarrow \Sigma_5(\mathbb{P}_4) \models t_1'(\boldsymbol{X}, \boldsymbol{x}) = t_2'(\boldsymbol{X}, \boldsymbol{x}),$$

where t_i' is obtained by regarding the process variables of t_i also as condition variables.

5 Generalization of ACP and CpSP

We discuss various systems that generalize $\mathsf{ACP}(A, \gamma)$ [10] to a setting in which alternative composition is a special case of conditional composition, and that provides a parameterized version of the parallel composition operations. Next we provide an algebraic setting for the Cooperating Sequential Processes (CpSP) of Dijkstra [15]. We can do this for all logics that contain C. We define the following operations, where A is the set of atomic actions, Pr is the sort of processes, and $\mathbb{L}$ is the particular logic involved.

— *Constants and operations* —	— *Parametrized operations* —
$a \quad : A \subseteq Pr$	$\_ +_{\_} \_ \; : Pr \times \mathbb{L} \times Pr \to Pr$
$\delta \quad : Pr, \; \delta \notin A$	$\_ \,_{\_}\|_{\_}\, \_ \; : Pr \times \mathbb{L} \times \mathbb{L} \times Pr \to Pr$
$\_ \mid \_ \quad : A_\delta \times A_\delta \to A_\delta$	$\_ \,_{\_}\|\!\!\__{\_}\, \_ \; : Pr \times \mathbb{L} \times \mathbb{L} \times Pr \to Pr$
$\_ \cdot \_ \quad : Pr \times Pr \to Pr$	$\_ \,_{\_}|_{\_}\, \_ \; : Pr \times \mathbb{L} \times \mathbb{L} \times Pr \to Pr$
$\partial_H(\_) : Pr \to Pr \quad (H \subseteq A)$	$\_ \,_{\_}\lfloor_{\_}\, \_ \; : Pr \times \mathbb{L} \times \mathbb{L} \times Pr \to Pr$

We write $G_k(Z)$ for the k-valued generalization of axiomatization Z. We first describe the simplest generalization

$$G_3\big(\mathsf{ACP}_{\mathsf{C}}(A, \gamma, \mathbb{P}_2)\big).$$

and write $\Sigma_3^C(\mathbb{P}_2)$ for three-valued logic over $\{\mathsf{C}, \mathsf{T}, \mathsf{F}\}$ and $\mathbb{P}_2$. The system $G_3\big(\mathsf{ACP}_{\mathsf{C}}(A, \gamma, \mathbb{P}_2)\big)$ is defined by the axioms in Table 6, where $\gamma = \mid\restriction_{(A \times A)}$. Observe that axiom (GA3) is equivalent with

$$X +_{\phi} X = X,$$

as $\mathsf{T} \triangleleft \phi \triangleright \mathsf{T} = \mathsf{T}$ in $\Sigma_3^C(\mathbb{P}_2)$. However, the formulation used in Table 6 allows straightforward generalizations to systems that contain error-prone conditions (possibly evaluating to M or D). It is easy to see which axioms should be added, e.g., if only D is considered, the axiom

$$\text{(GGD)} \quad X +_{\mathsf{D}} Y = \delta$$

should be added to Table 6. Involving M gives rise to $\mu \in Pr$ and axioms

$$\begin{array}{ll} \text{(GM1)} & X +_{\mathsf{C}} \mu = \mu, \\ \text{(GGM)} & X +_{\mathsf{M}} Y = \mu. \end{array}$$

Observe that $\mu X = \mu$ is derivable from (GGM) and (GA4). Furthermore, $X \,_{\phi}\lfloor_{\psi}\, \mu = \mu \,_{\phi}\lfloor_{\psi}\, X = \mu$ follows from (GGM) and (GCM8), (GCM9), respectively. The system

$$G_5\big(\mathsf{ACP}_{\mathsf{C},\mu}(A, \gamma, \mathbb{P}_4)\big)$$

is defined as the extension of $G_3\big(\mathsf{ACP}_{\mathsf{C}}(A, \gamma, \mathbb{P}_2)\big)$ with (GM1), (GGM), (GGD), and with conditions ranging over $\Sigma_5(\mathbb{P}_4)$.

Table 6. $G_3\Big(\mathsf{ACP}_{\mathsf{C}}(A, \gamma, \mathbb{P}_2)\Big)$, $a, b \in A_\delta$, $H \subseteq A$, and $\phi, \psi, \chi \in \Sigma_3^C(\mathbb{P}_2)$.

(GGT)	$X +_{\mathsf{T}} Y = X$
(GA1)	$X +_{\phi} (Y +_{\phi} Z) = (X +_{\phi} Y) +_{\phi} Z$
(GA2)	$X +_{\phi} Y = Y +_{\neg\phi} X$
(GA3)	$X +_{\phi} X = X +_{(\mathsf{T} \lhd \phi \rhd \mathsf{T})} X$
(GA4)	$(X +_{\phi} Y)Z = XZ +_{\phi} YZ$
(GA5)	$(XY)Z = X(YZ)$
(GA6)	$X +_{\mathsf{C}} \delta = X$
(GA7)	$\delta X = \delta$
(C1)	$a \mid b = b \mid a$
(C2)	$(a \mid b) \mid c = a \mid (b \mid c)$
(C3)	$\delta \mid a = \delta$
(GCM1)	$X \,_{\phi}\Vert_{\psi}\, Y = (X \,_{\phi}\llfloor_{\psi}\, Y +_{\psi} Y \,_{\phi}\llfloor_{\neg\psi}\, X) +_{\phi} X \,_{\phi}\vert_{\psi}\, Y$
(GCM2)	$a \,_{\phi}\llfloor_{\psi}\, X = aX$
(GCM3)	$aX \,_{\phi}\llfloor_{\psi}\, Y = a(X \,_{\phi}\Vert_{\psi}\, Y)$
(GCM4)	$(X +_{\phi} Y) \,_{\psi}\llfloor_{\chi}\, Z = X \,_{\psi}\llfloor_{\chi}\, Z +_{\phi} Y \,_{\psi}\llfloor_{\chi}\, Z$
(GCMC)	$X \,_{\phi}\vert_{\psi}\, Y = X \,_{\phi}\lfloor_{\psi}\, Y +_{\psi} Y \,_{\phi}\lfloor_{\neg\psi}\, X$
(GCM5)	$aX \,_{\phi}\lfloor_{\psi}\, Y = a \,_{\phi}\lfloor_{\psi}\, (Y \,_{\phi}\llfloor_{\neg\psi}\, X)$
(GCM6)	$a \,_{\phi}\lfloor_{\psi}\, b = a \mid b$
(GCM7)	$a \,_{\phi}\lfloor_{\psi}\, bX = (a \mid b)X$
(GCM8)	$a \,_{\phi}\lfloor_{\psi}\, (X +_{\chi} Y) = a \,_{\phi}\lfloor_{\psi}\, X +_{\chi} a \,_{\phi}\lfloor_{\psi}\, Y$
(GCM9)	$(X +_{\phi} Y) \,_{\psi}\lfloor_{\chi}\, Z = X \,_{\psi}\lfloor_{\chi}\, Z +_{\phi} Y \,_{\psi}\lfloor_{\chi}\, Z$
(GD1)	$\partial_H(a) = a \quad \text{if } a \notin H$
(GD2)	$\partial_H(a) = \delta \quad \text{if } a \in H$
(GD3)	$\partial_H(X +_{\phi} Y) = \partial_H(X) +_{\phi} \partial_H(Y)$
(GD4)	$\partial_H(XY) = \partial_H(X)\partial_H(Y)$

Cooperating Sequential Processes, CpSP, in the style of [15] can be abstractly modeled in $G_5\big(\mathsf{PA}_{\delta,\mathsf{C},\mu}(A, \mathbb{P}_4)\big)$ with action history operator and state operator. Here, $\mathsf{PA}_{\delta,\mathsf{C},\mu}(A, \mathbb{P}_4)$ refers to the restriction of parallel composition to interleaving, thus to a setting without communication, and is obtained from

$G_5\big(\mathsf{ACP}_{\mathsf{C},\mu}(A,\gamma,\mathbb{P}_4)\big)$ by restricting ${}_\phi\|_\psi$ to ${}_\mathsf{T}\|_\mathsf{C}$. We further write $\|$ for ${}_\mathsf{T}\|_\mathsf{C}$, and $\mathbin{|\!\!\lfloor}$ instead of ${}_\mathsf{T}\mathbin{|\!\!\lfloor}_\mathsf{C}$. The axioms of $G_5\big(\mathsf{PA}_{\delta,\mathsf{C},\mu}(A,\mathbb{P}_4)\big)$ are given in Table 7.

Table 7. $G_5\Big(\mathsf{PA}_{\delta,\mathsf{C},\mu}(A,\mathbb{P}_4)\Big)$, $a\in A_\delta\cup\{\mu\}$, $\sigma\in A^*$, and $\phi\in\Sigma_5(\mathbb{P}_4)$.

(GA1)	$X+_\phi(Y+_\phi Z)=(X+_\phi Y)+_\phi Z$	(GGT)	$X+_\mathsf{T}Y=X$
(GA2)	$X+_\phi Y=Y+_{\neg\phi}X$	(GGD)	$X+_\mathsf{D}Y=\delta$
(GA3)	$X+_\phi X=X+_{(\mathsf{T}\lhd\phi\rhd\mathsf{T})}X$	(GM1)	$X+_\mathsf{C}\mu=\mu$
(GA4)	$(X+_\phi Y)Z=XZ+_\phi YZ$	(GGM)	$X+_\mathsf{M}Y=\mu$
(GA5)	$(XY)Z=X(YZ)$		
(GA6)	$X+_\mathsf{C}\delta=X$		
(GA7)	$\delta X=\delta$		
(GCM1)	$X\parallel Y=X\mathbin{|\!\!\lfloor}Y+_\mathsf{C}Y\mathbin{|\!\!\lfloor}X$		
(GCM2)	$a\mathbin{|\!\!\lfloor}X=aX$		
(GCM3)	$aX\mathbin{|\!\!\lfloor}Y=a(X\parallel Y)$		
(GCM4)	$(X+_\phi Y)\mathbin{|\!\!\lfloor}Z=X\mathbin{|\!\!\lfloor}Z+_\phi Y\mathbin{|\!\!\lfloor}Z$		

Action History Logic, AHL, was introduced in [13] as a natural example of the use of four-valued logic in process algebra. It can be used to express history dependent properties of processes, and comprises the following ingredients:

In, the assertion which is true of the initial state of a process and false thereafter.
$\mathsf{P}_4(\phi)$, the assertion that ϕ is valid in the previous state, i.e., the state before the last action. If there is no such state, $\mathsf{P}_4(\phi)=\mathsf{M}$.
$\mathsf{L}_4(a)$, the condition that expresses that the last action was a. In case the state is initial, $\mathsf{L}_4(a)$ evaluates to M.

Let $\Sigma_5(\mathbb{P}_4)$ be generated from AHL. Writing ϵ for the empty history, the *action history operator* H_ϵ defined in Table 8 memorizes the action history (trace) of a fluent-free process. In order to represent a CpSP-process which involves the interpretation of fluents we consider a data-state space $\mathcal{S}\subseteq\mathcal{T}\times\mathcal{W}$ for some further unspecified set $\mathcal{T}$ and the set $\mathcal{W}$ of interpretations. We use a *state operator* $\lambda_s(\_)$ (see [2]) to model how the execution of actions affects interpretations. Typically, process aX in data-state s is represented as $\lambda_s(aX)$ and satisfies

$$\lambda_s(aX)=a'\cdot\lambda_{s'}(X)$$

where a' is the action (or δ or μ) that occurs as the result of executing a in data-state s, and s' is the data-state which ensues when executing a in s. We assume two given functions describing these effects: $\mathit{action}: A\times\mathcal{S}\to A\cup\{\delta,\mu\}$ and $\mathit{effect}: A\times\mathcal{S}\to\mathcal{S}$. We further set $\mathit{action}(c,s)=c$ for $c\in\{\delta,\mu\}$. Axioms for the state operator are also given in Table 8.

Now $H_\epsilon(\lambda_s(P_1\parallel ...\parallel P_n))$ with P_i not containing history/state operators or $\parallel$, $\mathbin{|\!\!\lfloor}$ typically is an algebraic notation for a CpSP-process with (global) initial data-state s.

Table 8. Axioms for history and state operator, $a \in A$, $\sigma \in A^*$, and $\phi, \psi \in \Sigma_5(\mathbb{P}_4)$.

$H_\sigma(X +_\phi Y) = H_\sigma(X) +_{\phi(\sigma)} H_\sigma(Y)$ $H_\sigma(c) = c$ for $c \in A \cup \{\delta, \mu\}$ $H_\sigma(a \cdot X) = a \cdot H_{\sigma a}(X)$ $c(\sigma) = c$ for $c \in \{\mathsf{M}, \mathsf{C}, \mathsf{T}, \mathsf{F}, \mathsf{D}\}$ $(\neg\phi)(\sigma) = \neg(\phi(\sigma))$ $(\phi \diamond \psi)(\sigma) = \phi(\sigma) \diamond \psi(\sigma)$ for $\diamond \in \{\wedge, {\circ}\!\!\wedge\}$	$\mathsf{In}(\epsilon) = \mathsf{T}$ $\mathsf{In}(\sigma a) = \mathsf{F}$ $\mathsf{P}_4(\phi)(\epsilon) = \mathsf{M}$ $\mathsf{P}_4(\phi)(\sigma a) = \phi(\sigma)$ $\mathsf{L}_4(a)(\epsilon) = \mathsf{M}$ $\mathsf{L}_4(a)(\sigma b) = a \equiv b \in \{\mathsf{T}, \mathsf{F}\}$
$\lambda_{(t,w)}(X +_\phi Y) = \lambda_{(t,w)}(X) +_{w(\phi)} \lambda_{(t,w)}(Y)$ $\lambda_s(c) = action(c, s)$ for $c \in A \cup \{\delta, \mu\}$	$\lambda_s(aX) = action(a, s) \cdot \lambda_{s'}(X)$ where $s' = effect(a, s)$

6 Conclusions

We observed that Kleene's three-valued logic $\mathbb{K}_3$ allows for two intuitions of the third, non-classical truth value: undetermined and undefined. Indeed, a complete axiomatization of $\mathbb{K}_3$ leaves room for exactly two non-classical constants, notation C and D, and implies $\mathsf{C} \wedge \mathsf{D} = \mathsf{F}$. The resulting four-valued logic $\mathbb{K}_4$ has a complete, equational axiomatization [20]. The combination of $\mathbb{K}_4$, or one of its sublogics containing T, F, with process algebra yields an equational completeness result (adopting our restriction on the interpretation of fluents, discarding C, and using Lemma 4.3). This follows from [4, 12]. Adding M (meaningless) to $\mathbb{K}_4$ yields a five-valued logic, which we extended with McCarthy's asymmetric connectives to provide a useful combination with process algebra. We presented a non-equational completeness result (using the 'excluded fifth rule'). Completeness results for all sublogics containing $\mathsf{M}, \mathsf{T}, \mathsf{F}$ follow from [11, 12].

We hope to have indicated that the use of non-classical logics in process theory is interesting in its own right. Expressivity can be enlarged by involving recursive ingredients. For process description we propose the (binary) Kleene star (see [19]), which in process algebra is defined by $X^*Y = X \cdot (X^*Y) + Y$ (see also [7]). In a more general setting, one can define $X^{*\phi}Y = X \cdot (X^{*\phi}Y) +_\phi Y$ and write X^*Y for $X^{*\mathsf{C}}Y$. Examples with recursively defined conditions, such as *schedulers*, are discussed in [13].

References

1. K.R. Apt. Ten years of Hoare's logic, a survey, part I. *ACM Transactions on Programming Languages and Systems*, 3(4):431-483, 1981.
2. J.C.M. Baeten and J.A. Bergstra. Global renaming operators in concrete process algebra. *Information and Computation*, 78(3):205-245, 1988.
3. J.C.M. Baeten and J.A. Bergstra. Process algebra with signals and conditions. In M. Broy, editor, Programming and Mathematical Method, *Proceedings Summer School Marktoberdorf*, 1990 NATO ASI Series F, pages 273-323, Springer-Verlag, 1992.
4. J.C.M. Baeten and J.A. Bergstra. Process algebra with propositional signals. *Theoretical Computer Science*, 177(2):381-406, 1997.

5. J.C.M. Baeten, J.A. Bergstra, and S.A. Smolka. Axiomatizing probabilistic processes: ACP with generative probabilities. *Information and Computation*, 121(2):234-254, 1995.
6. J.C.M. Baeten and W.P. Weijland. *Process Algebra*. Cambridge Tracts in Theoretical Computer Science 18. Cambridge University Press, 1990.
7. J.A. Bergstra, I. Bethke, and A. Ponse. Process algebra with iteration and nesting. *Computer Journal*, 37(4):243-258, 1994.
8. J.A. Bergstra, I. Bethke, and P.H. Rodenburg. A propositional logic with 4 values: true, false, divergent and meaningless. *Journal of Applied Non-Classical Logics*, 5:199-217, 1995.
9. J.A. Bergstra and J.W. Klop. The algebra of recursively defined processes and the algebra of regular processes. In A. Ponse, C. Verhoef, and S.F.M. van Vlijmen, *Algebra of Communicating Processes, Utrecht 1994*, Workshops in Computing, pages 1-25. Springer-Verlag, 1995. An extended abstract appeared in J. Paredaens, editor, *Proceedings* 11^{th} *ICALP,* Antwerp, volume 172 of *Lecture Notes in Computer Science*, pages 82-95. Springer-Verlag, 1984.
10. J.A. Bergstra and J.W. Klop. Process algebra for synchronous communication. *Information and Control*, 60(1/3):109-137, 1984.
11. J.A. Bergstra and A. Ponse. Bochvar-McCarthy logic and process algebra. Technical Report P9722, Programming Research Group, University of Amsterdam, 1997 (see also `http://www.wins.uva.nl/research/prog/reports/reports.html`).
12. J.A. Bergstra and A. Ponse. Kleene's three-valued logic and process algebra. *Information Processing Letters*, 67(2):95-103, 1998.
13. J.A. Bergstra and A. Ponse. Process algebra with four-valued logic. Technical Report P9724, Programming Research Group, University of Amsterdam, 1997 (see also `http://www.wins.uva.nl/research/prog/reports/reports.html`). To appear in *Journal of Applied Non-Classical Logics*.
14. D.A. Bochvar. On a 3-valued logical calculus and its application to the analysis of contradictions (in Russian). *Matématičeskij sbornik*, 4:287-308, 1939.
15. E.W. Dijkstra. Cooperating sequential processes. In F. Genuys, editor, *Programming Languages*, pages 43-112, Academic Press, New York, 1968.
16. J.F. Groote. Transition system specifications with negative premises. *Theoretical Computer Science*, 118(2):263-299, 1993.
17. C.A.R. Hoare, I.J. Hayes, He Jifeng, C.C. Morgan, A.W. Roscoe, J.W. Sanders, I.H. Sorensen, J.M. Spivey, and B.A. Sufrin. Laws of programming. *Communications of the ACM*, 30(8):672-686, August 1987.
18. S.C. Kleene. On a notation for ordinal numbers. *Journal of Symbolic Logic*, 3:150-155, 1938.
19. S.C. Kleene. Representation of events in nerve nets and finite automata. In *Automata Studies*, pages 3-41. Princeton University Press, Princeton NJ, 1956.
20. S.P. Luttik and P.H. Rodenburg. Personal communications, 1998.
21. J. McCarthy. A basis for a mathematical theory of computation. In P. Braffort and D. Hirshberg (eds.), *Computer Programming and Formal Systems*, pages 33-70, North-Holland, Amsterdam, 1963.
22. J. McCarthy. *Formalization of common sense, papers by John McCarthy edited by V. Lifschitz*. Ablex, 1990.
23. J. McCarthy and P.J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463-502. Edinburgh University Press, 1969. Reprinted in [22].
24. C. Verhoef. A congruence theorem for structured operational semantics with predicates and negative premises. *Nordic Journal of Computing*, 2(2):274-302, 1995.